

# Main Architecture

Main Approach

104

105

1

1 of 39 drawings

Register Data Bus Controller

103

102

 $RO0 := RI0 + RI1$  $RO1 := \sim RI0$  $RO3 := -RI0$  $RO2 := \sim RI1$  $RO4 := -RI1$ 

Output Registers

RO0  
RO1  
RO2  
RO3  
RO4

FU00

Logic (Black box)

Input Registers

RI0  
RI1

101

Output Registers

RO5  
RO6  
RO7  
RO8  
RO9

FU01

Logic (Black box)

Input Registers

RI2  
RI3

FU02

Logic (Black box)

Input Registers

RI4

Output Registers

RO10  
RO11  
RO12  
RO13  
RO14

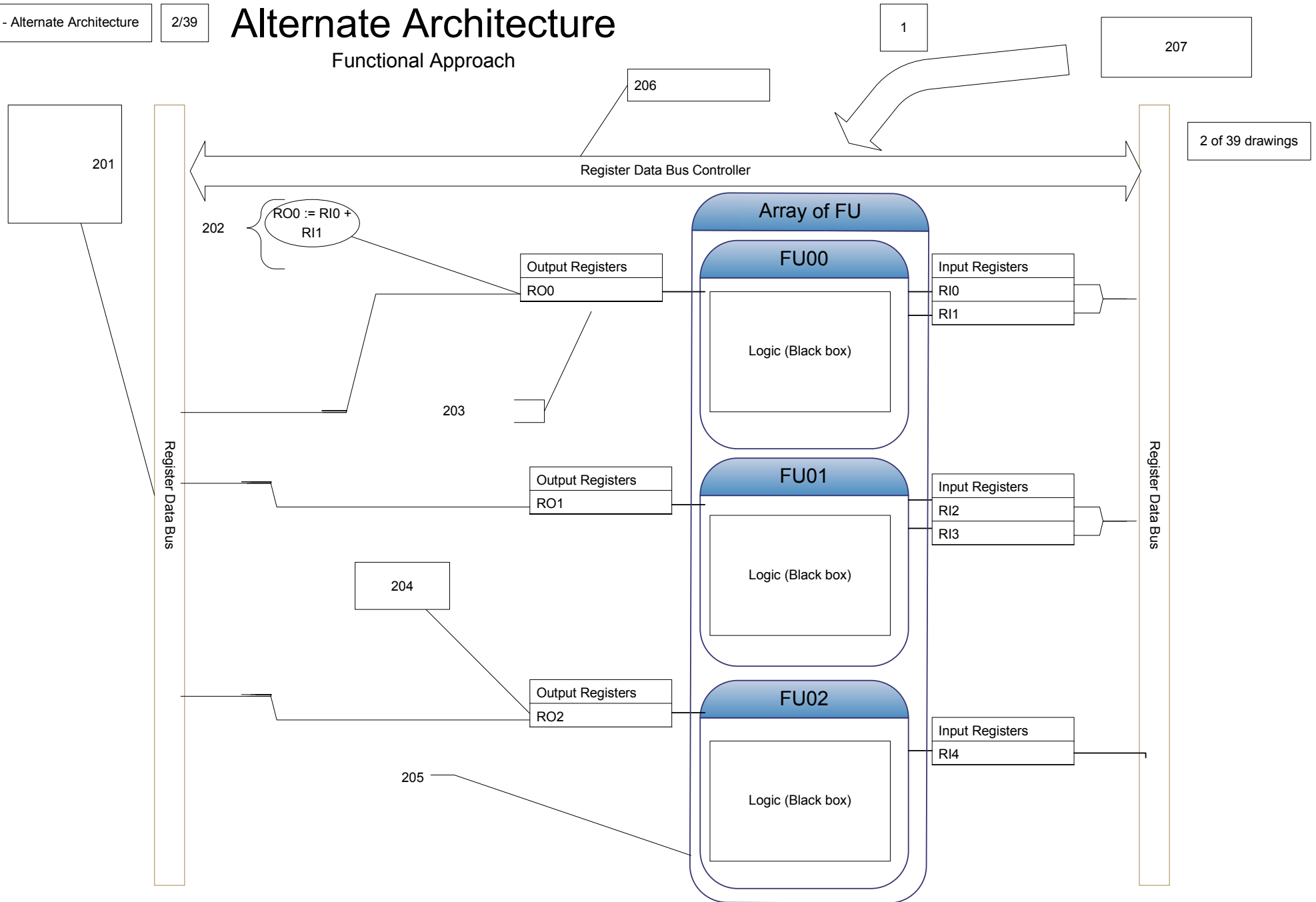
Register Data Bus

Register Data Bus

106

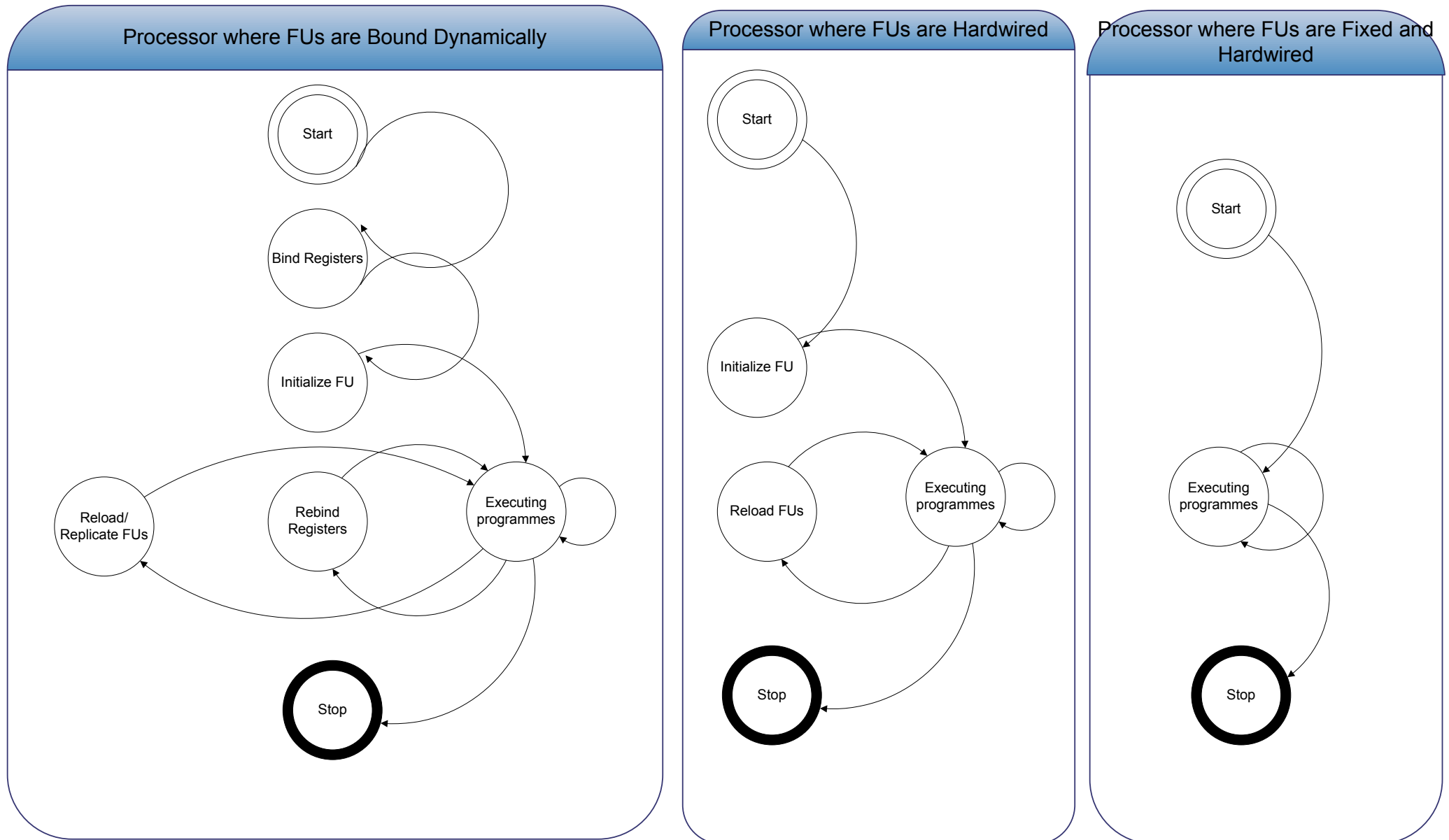
# Alternate Architecture

## Functional Approach



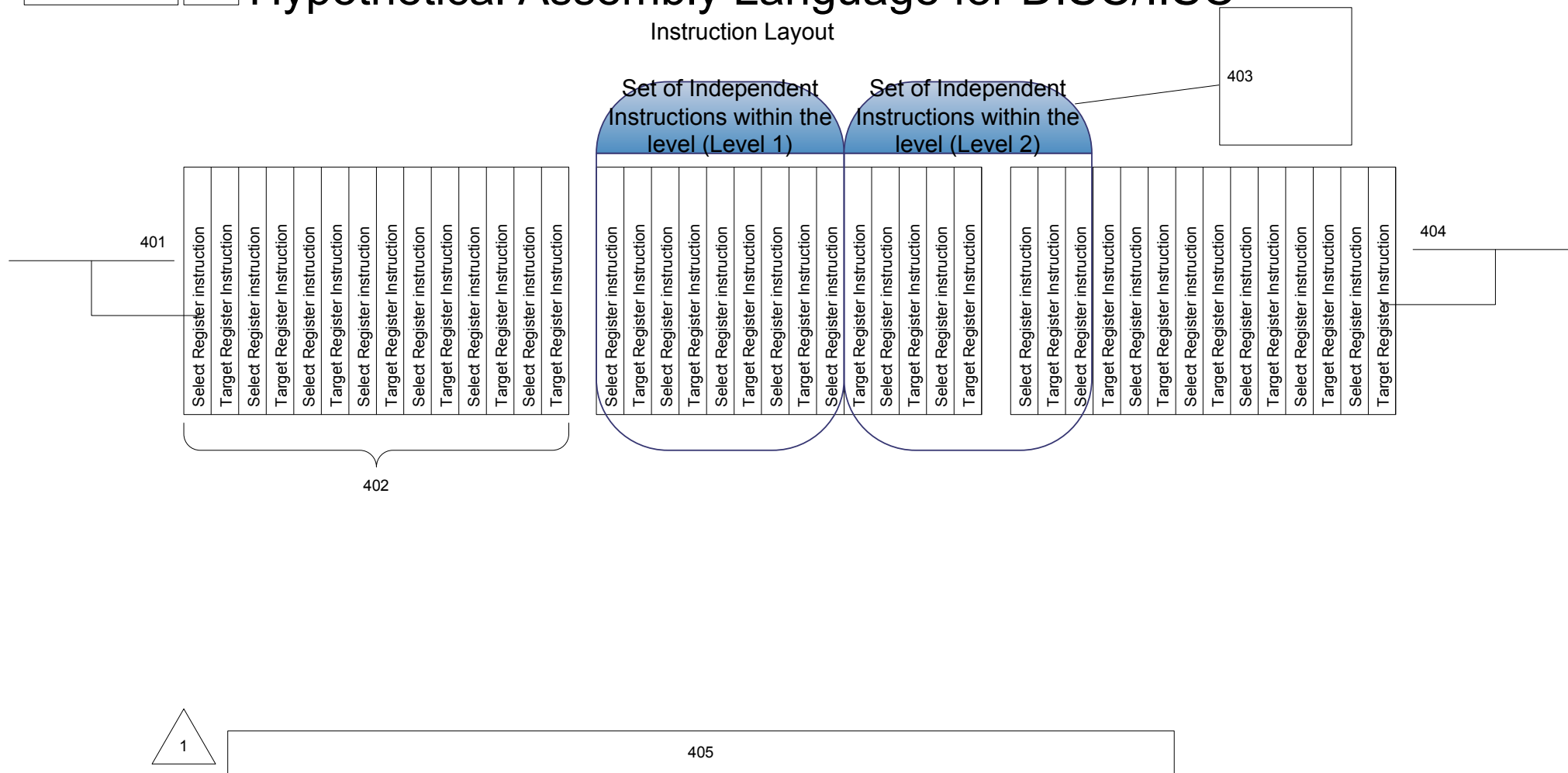
# Processor States

States of the Different Types of Processors that use DISA/IISA



# Hypothetical Assembly Language for DISC/IISC

## Instruction Layout



# Alternate Assembly Layouts

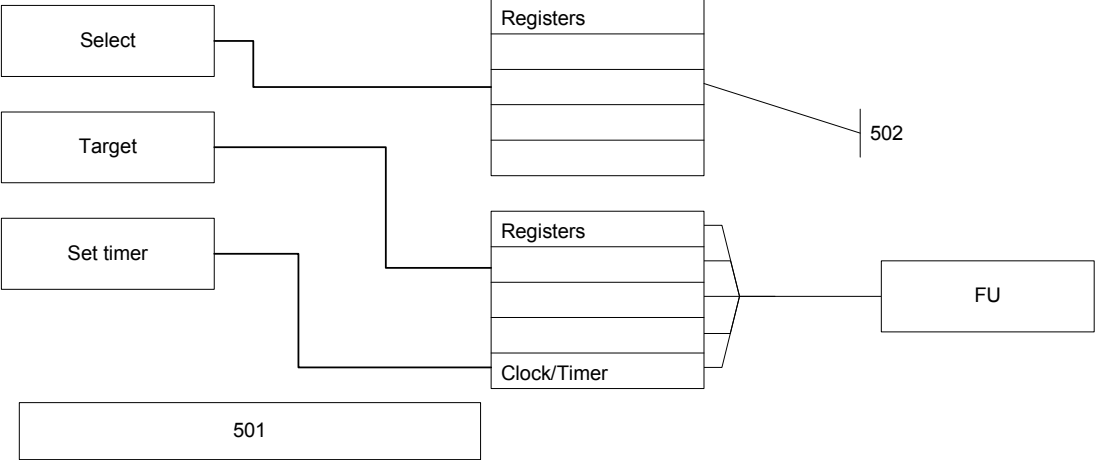
Fig: 5 - Alternate Instruction Layouts

5/39

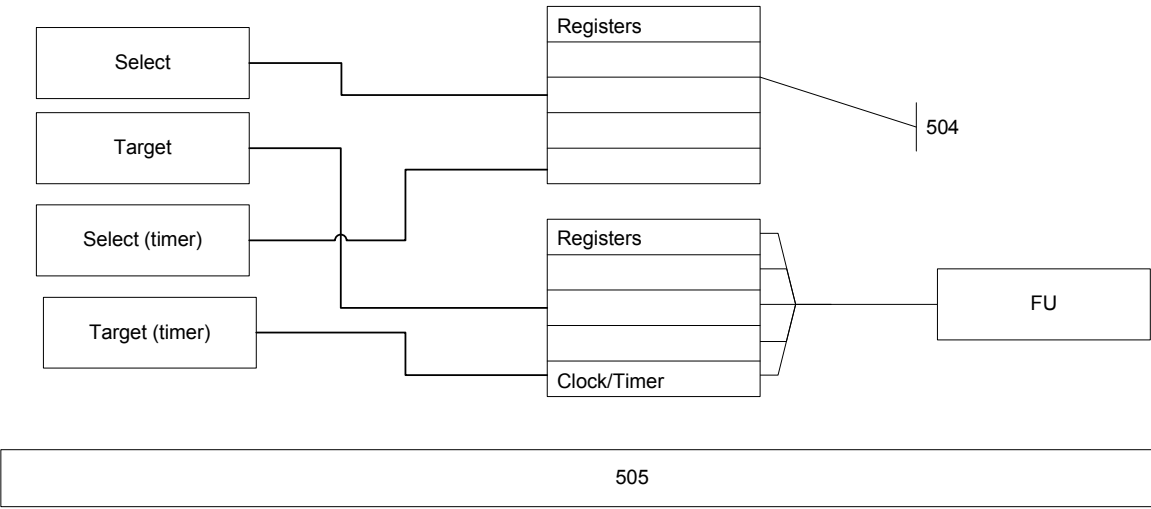
Instruction Layout

5 of 39 drawings

## Multiple Instruction Variation with a FU Scheduling Instruction



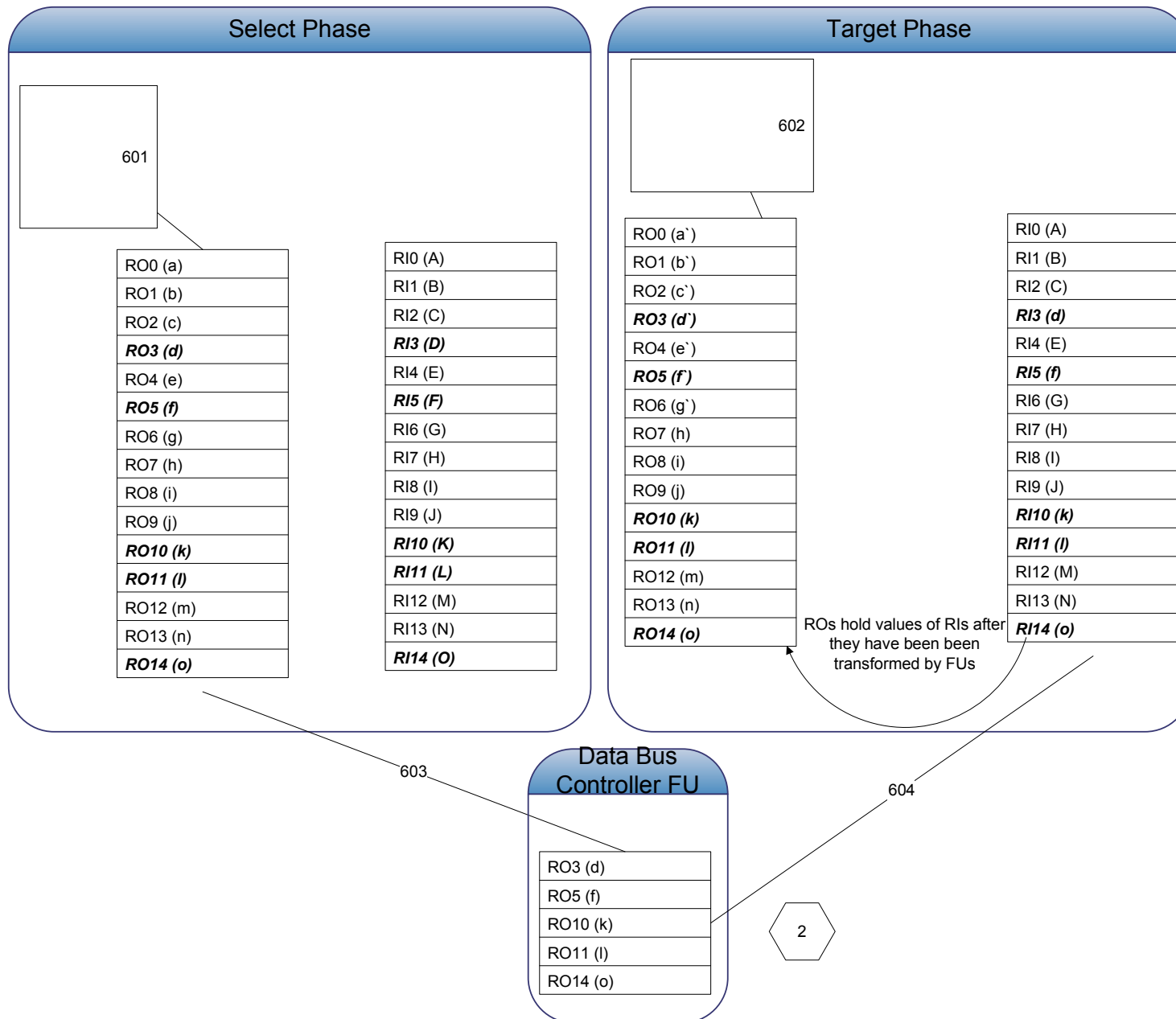
## Multiple Instruction Variation with a FU Scheduling by Updating the FU's Timer/scheduler



# Register Routing

Shows the two phases of execution in DISC/IISC

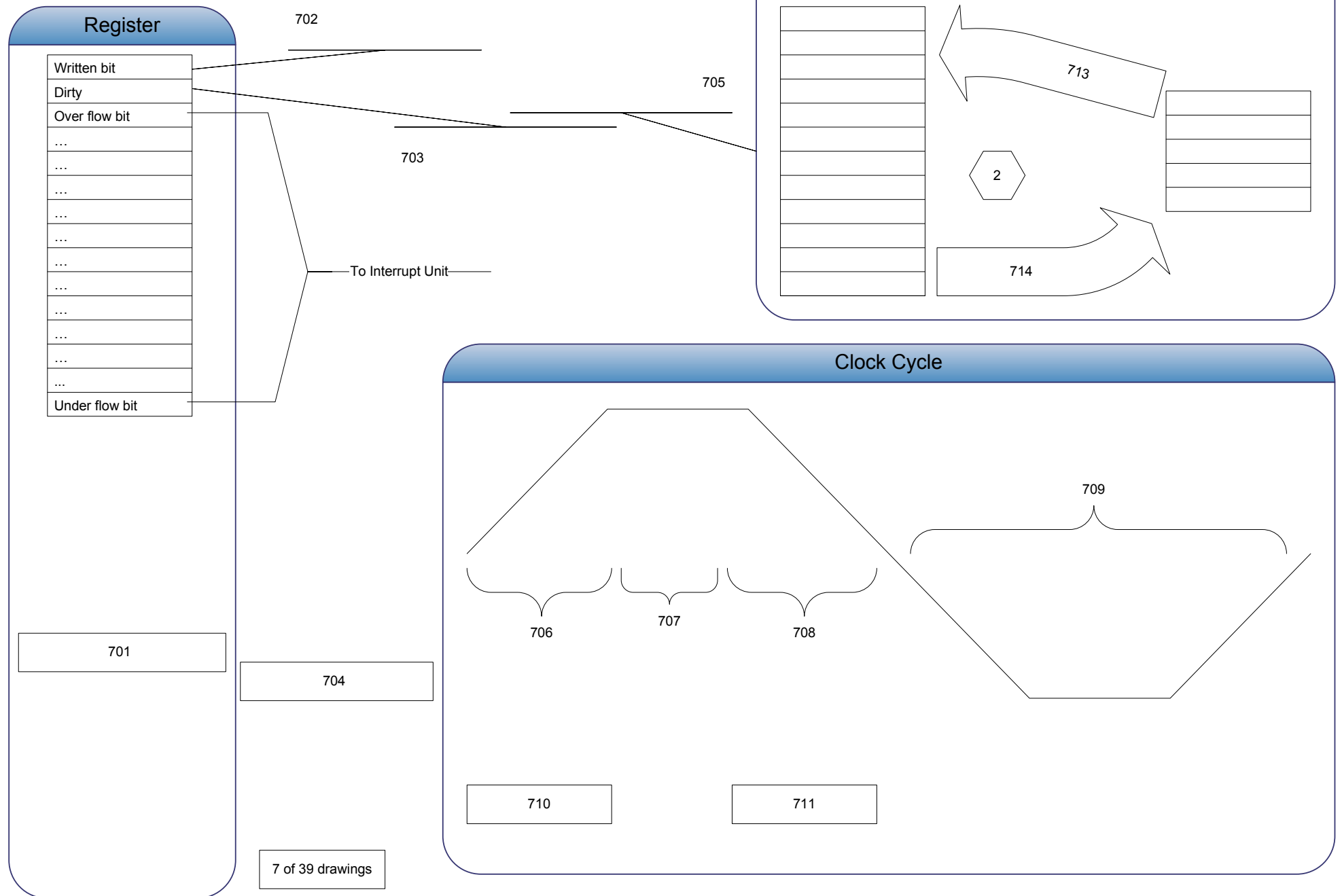
6 of 39 drawings



# Register Access

Fig: 7 - Register Access

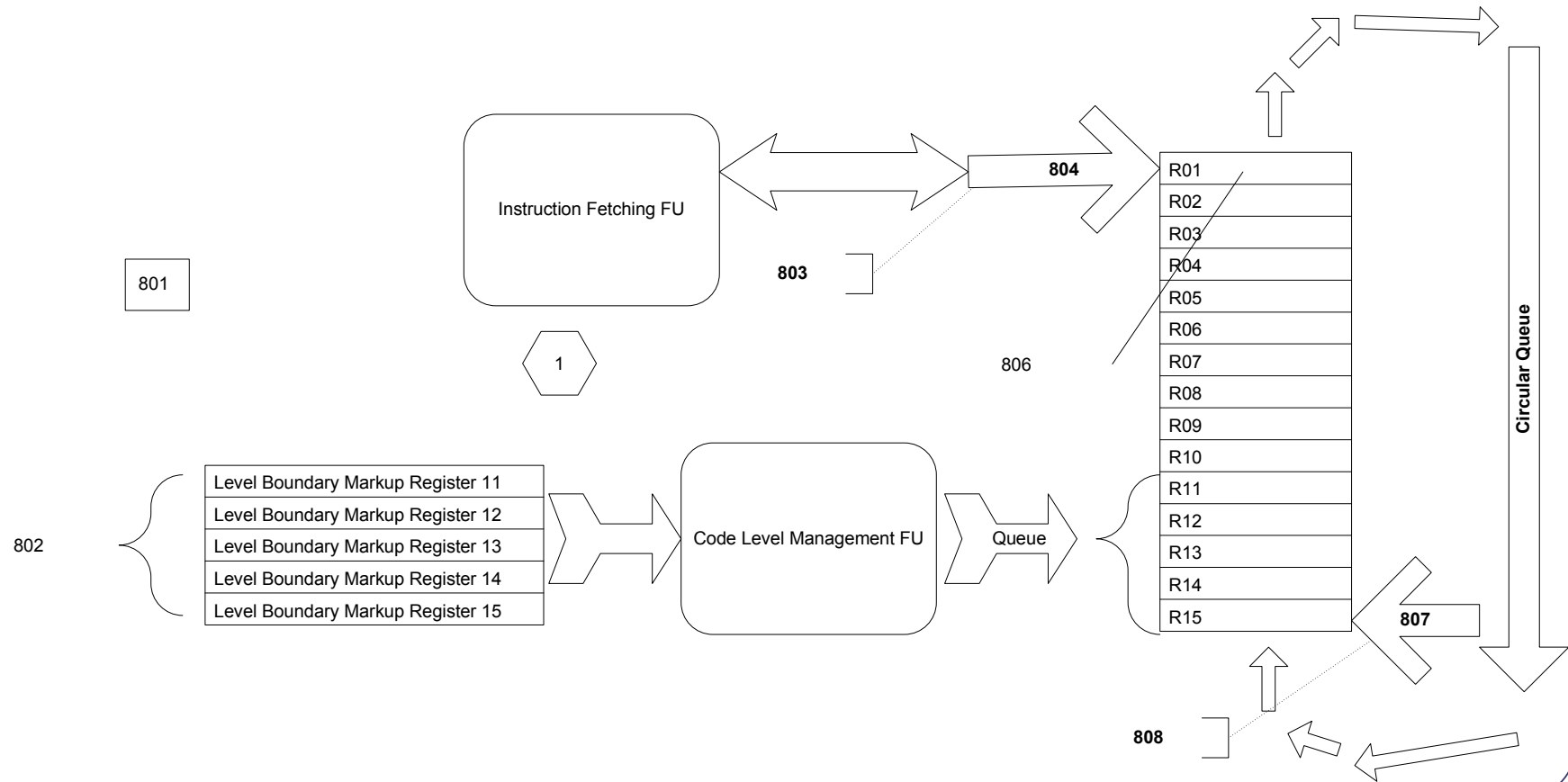
7/39



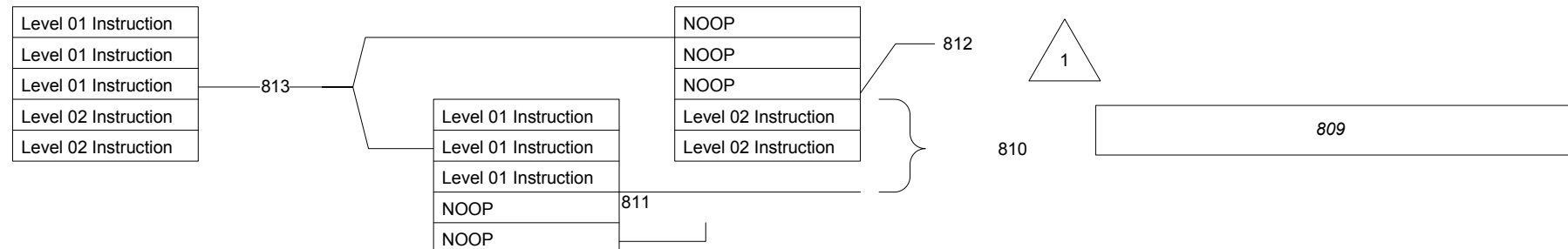
# Code Level Management Unit

Implementation and Concepts

## FU Layout



## Code Layout





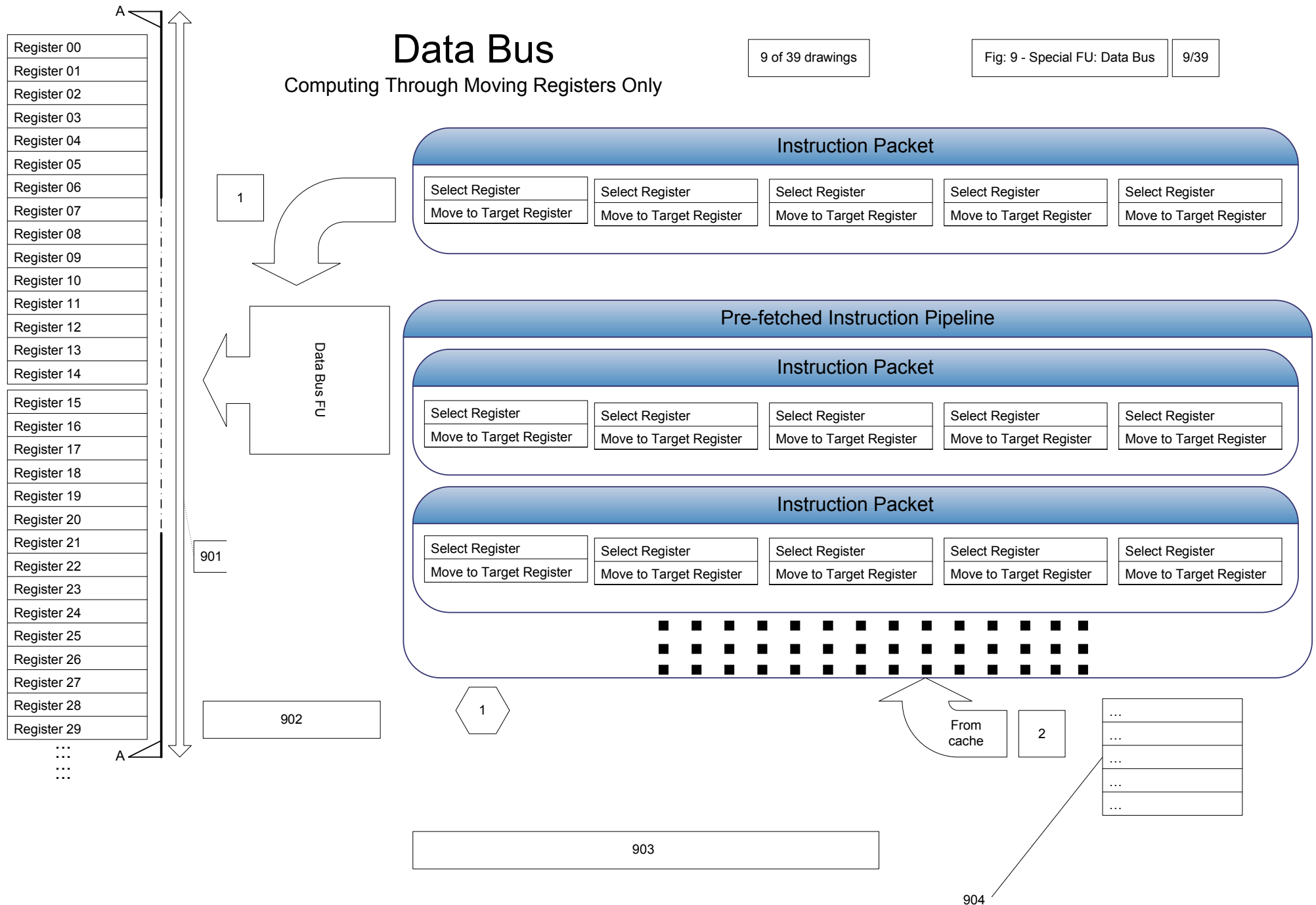
# Data Bus

9 of 39 drawings

Fig: 9 - Special FU: Data Bus

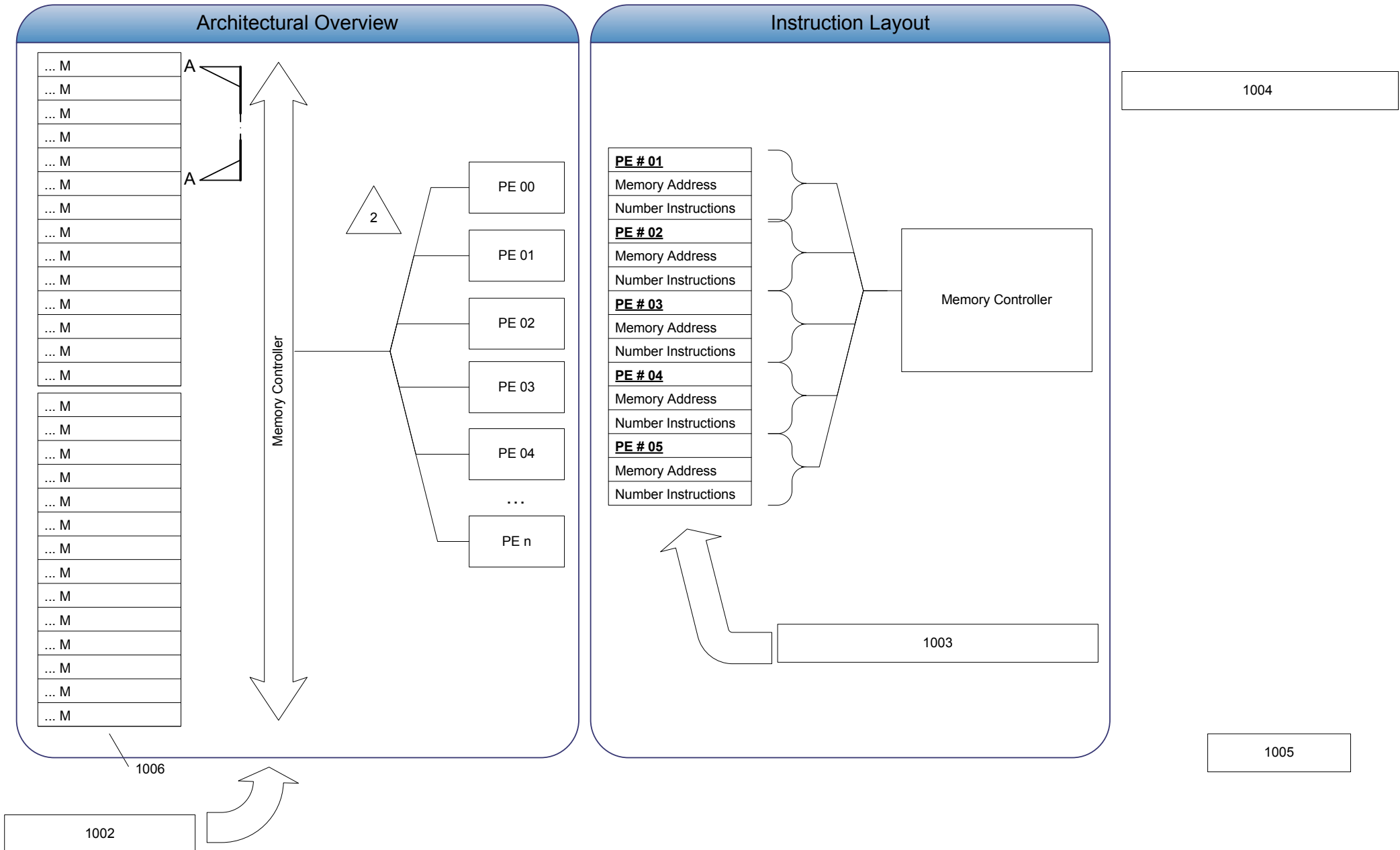
9/39

Computing Through Moving Registers Only



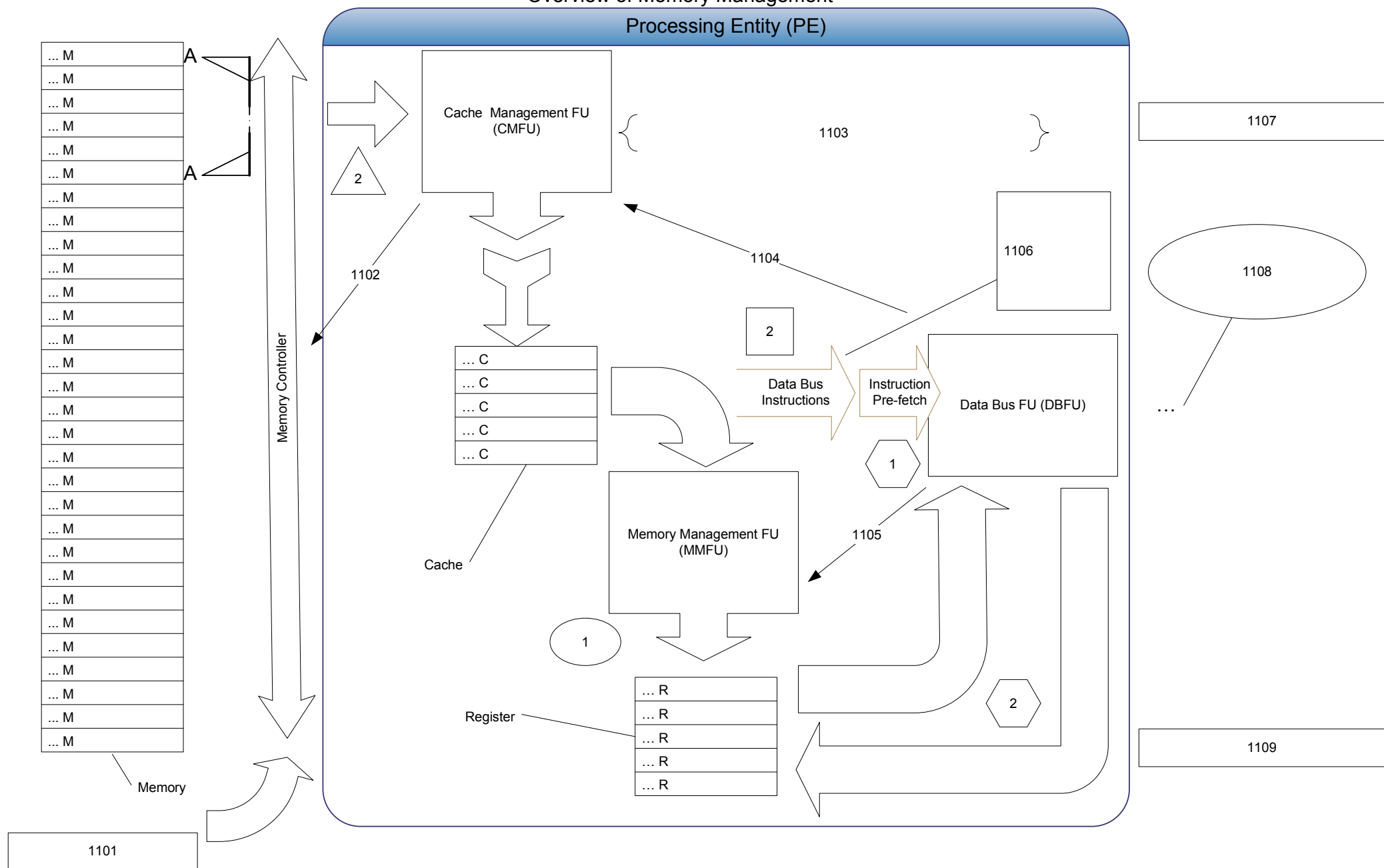
# Memory Controller

Transfer of Data and Instructions to PE



# Memory Management

## Overview of Memory Management



# Memory Controller

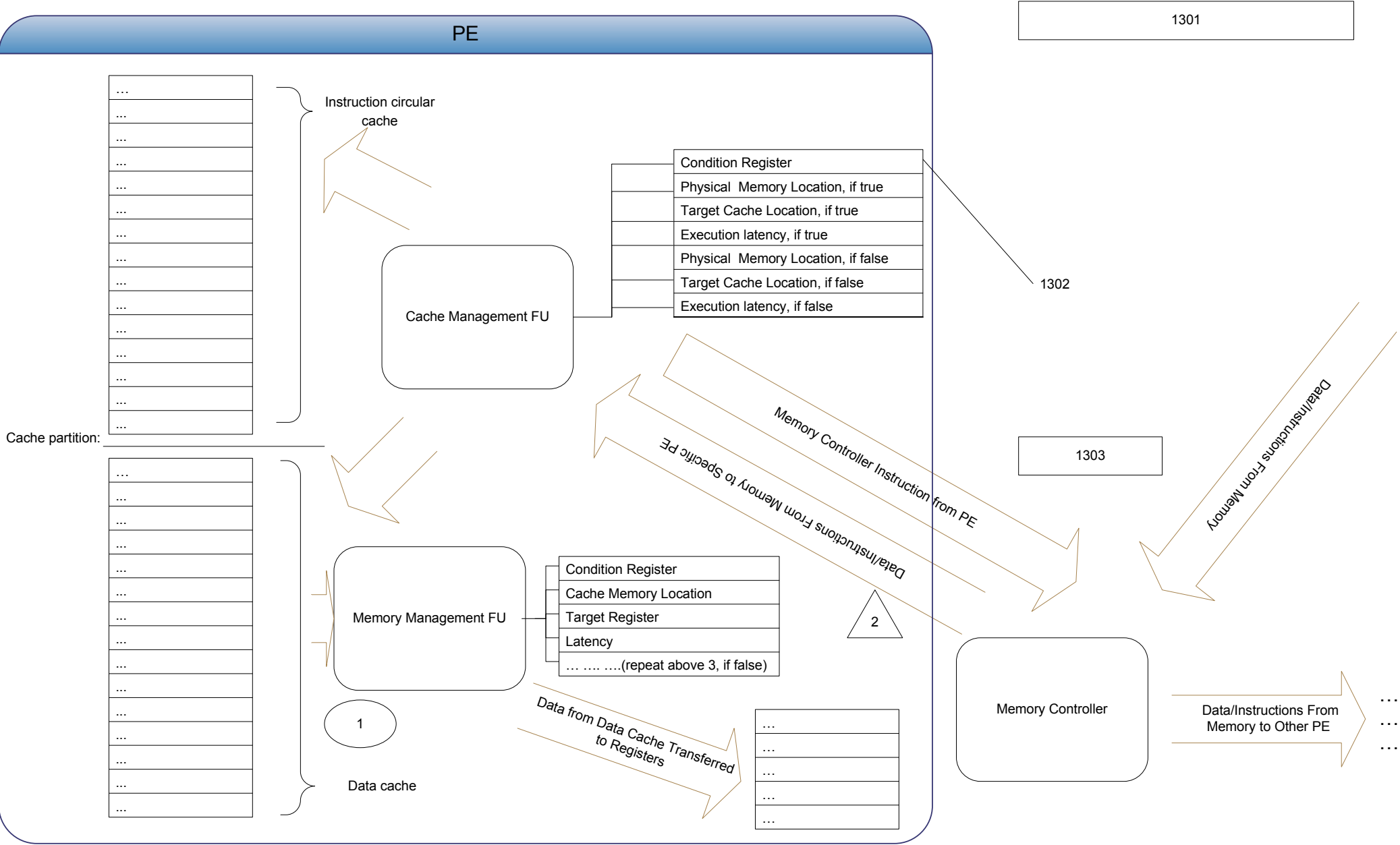
## Memory Management in Machines with Multiple PE



# Explicit Catch Management

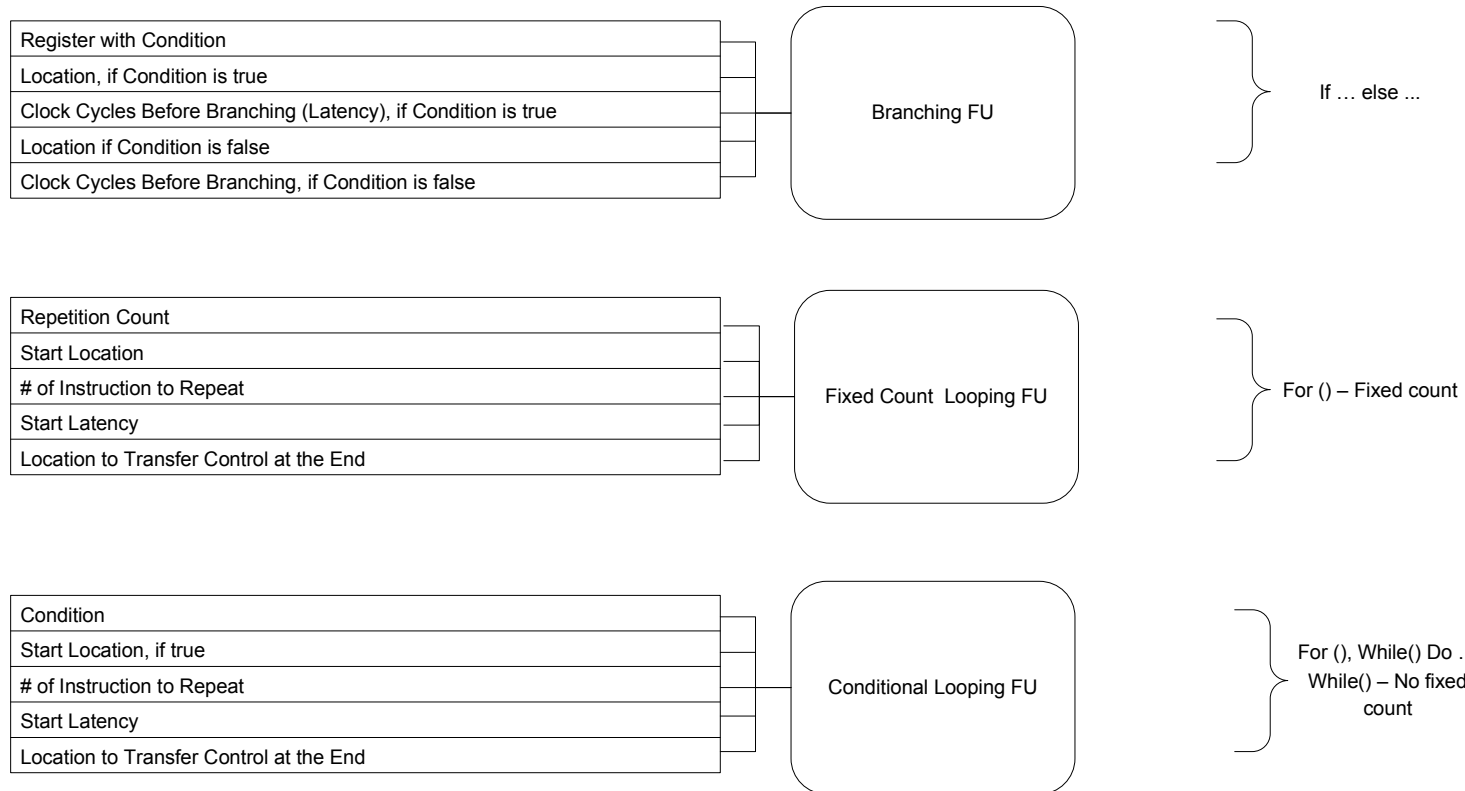
A Compiler Centric Cache Management Architecture

Fig: 13 - Special FU: Explicit Catch Management



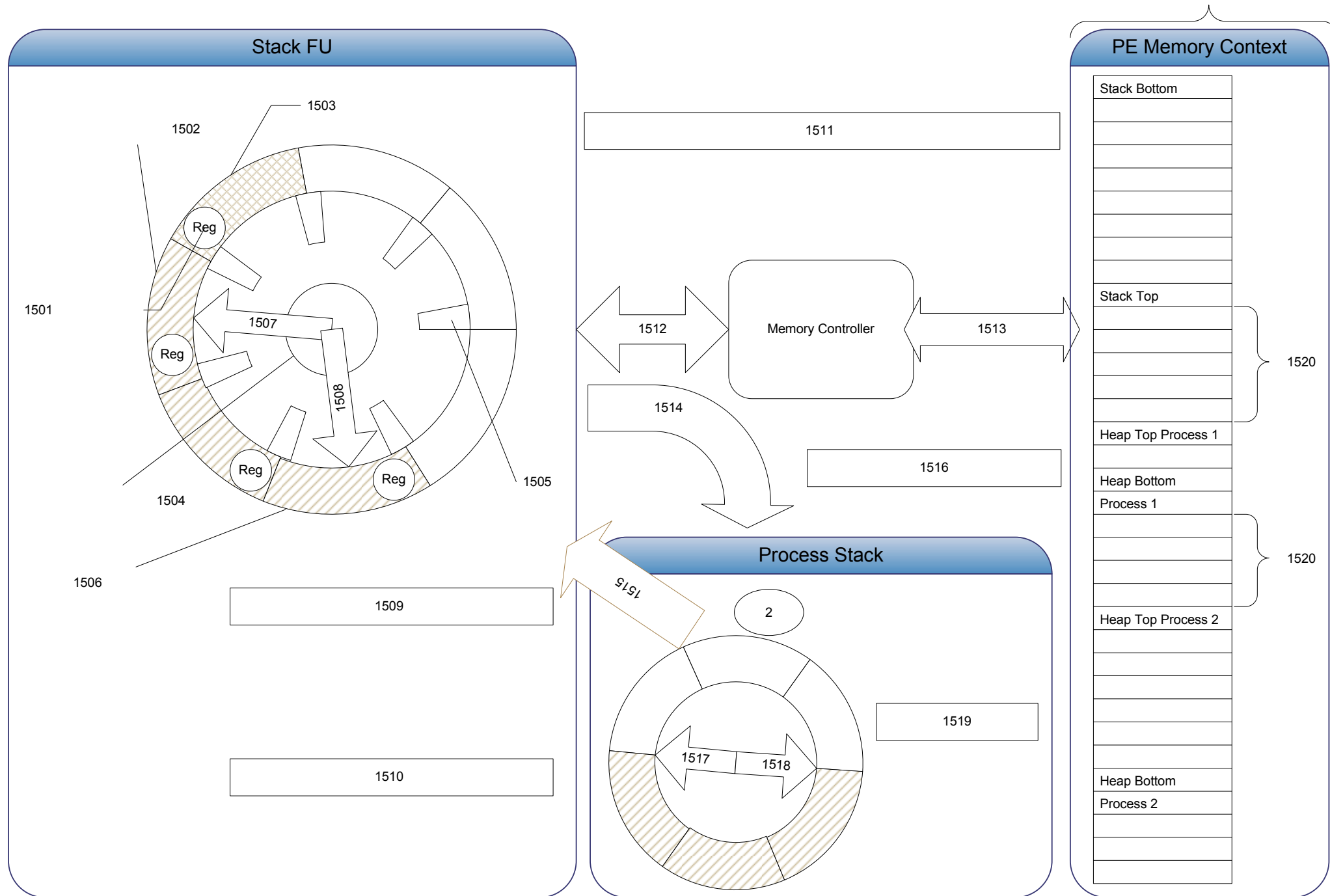
# Looping and Branching

## Scheduled Looping and Branching Architecture



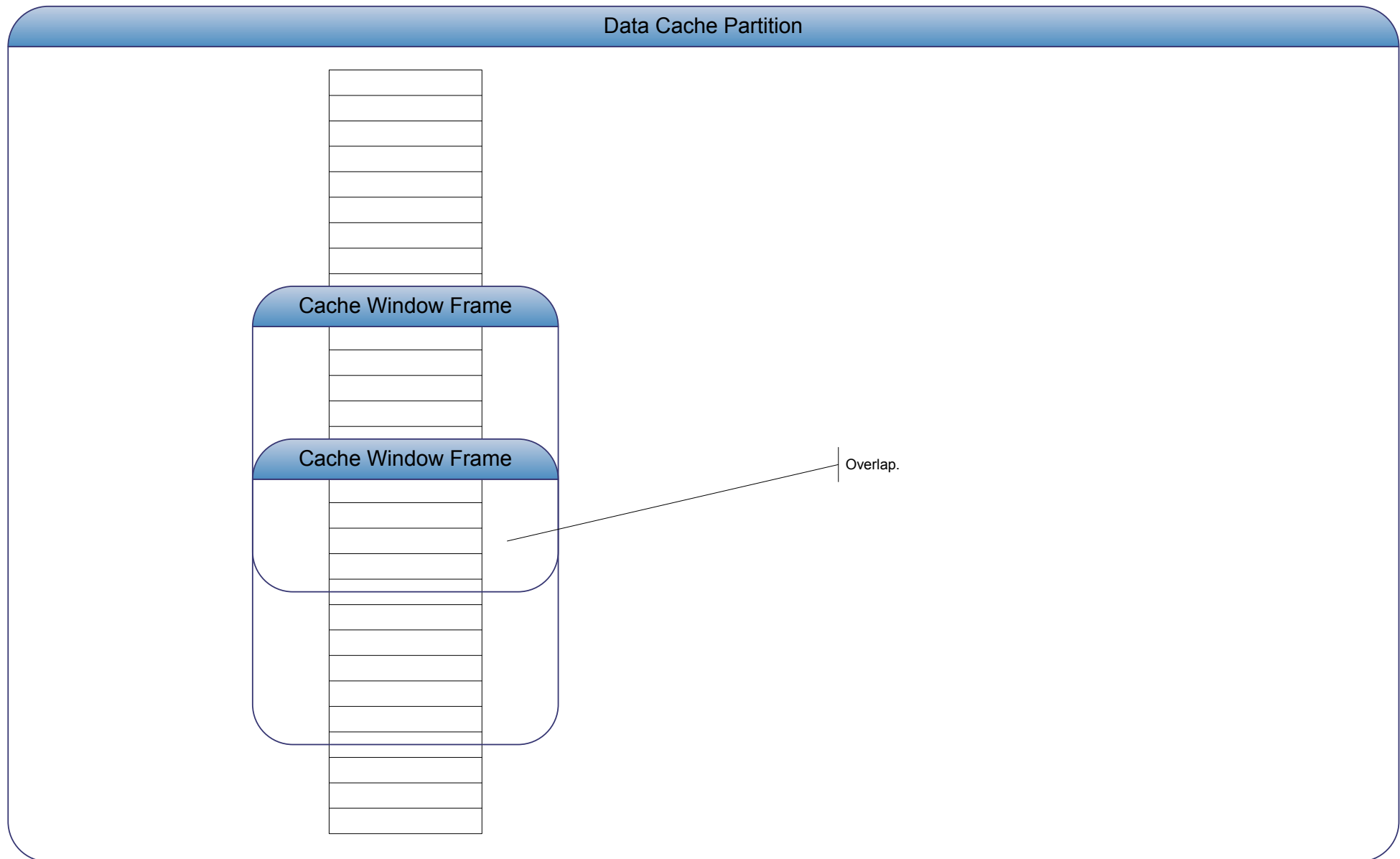
# Stack and Call Management

Repeated by the number of PE.



# Cache Window Frame in Function Calls

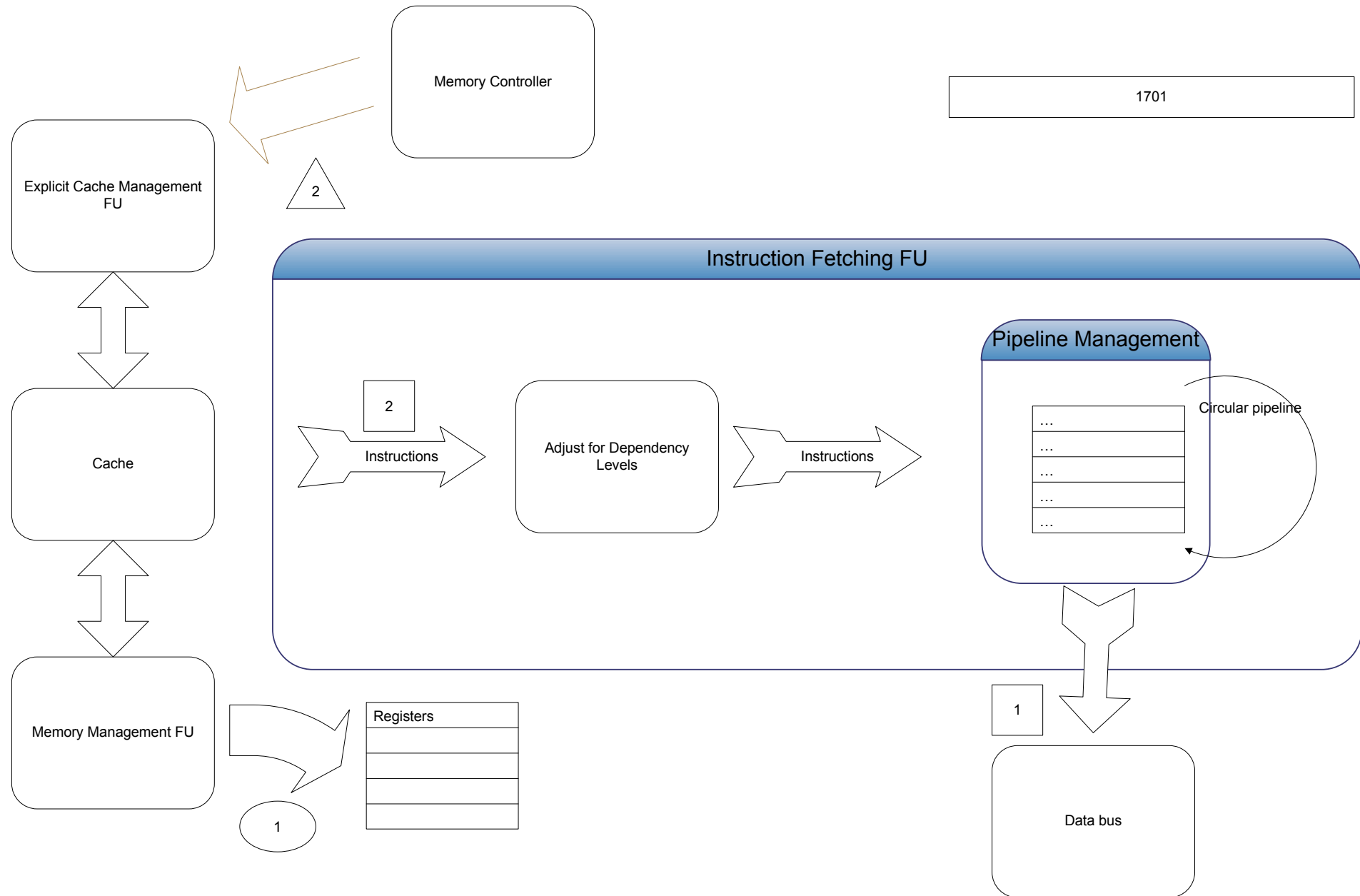
The change of cache addressing frame of reference on a function call





# Instruction Fetching FU

Fetching and Pipeline Management Unit

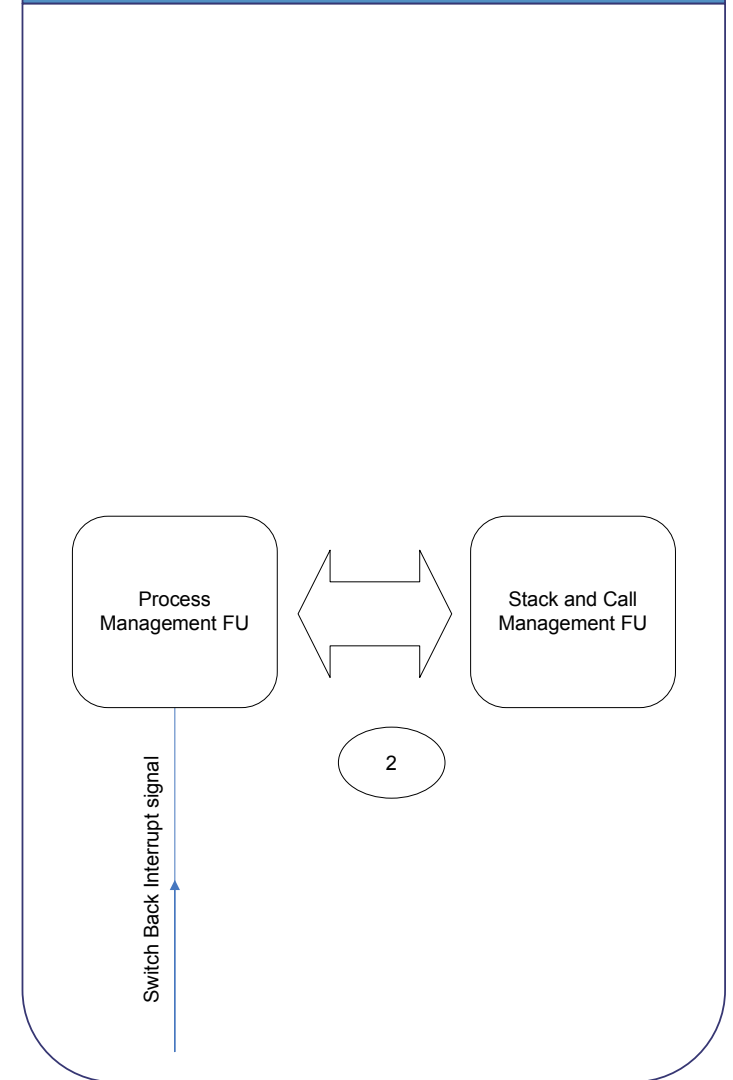
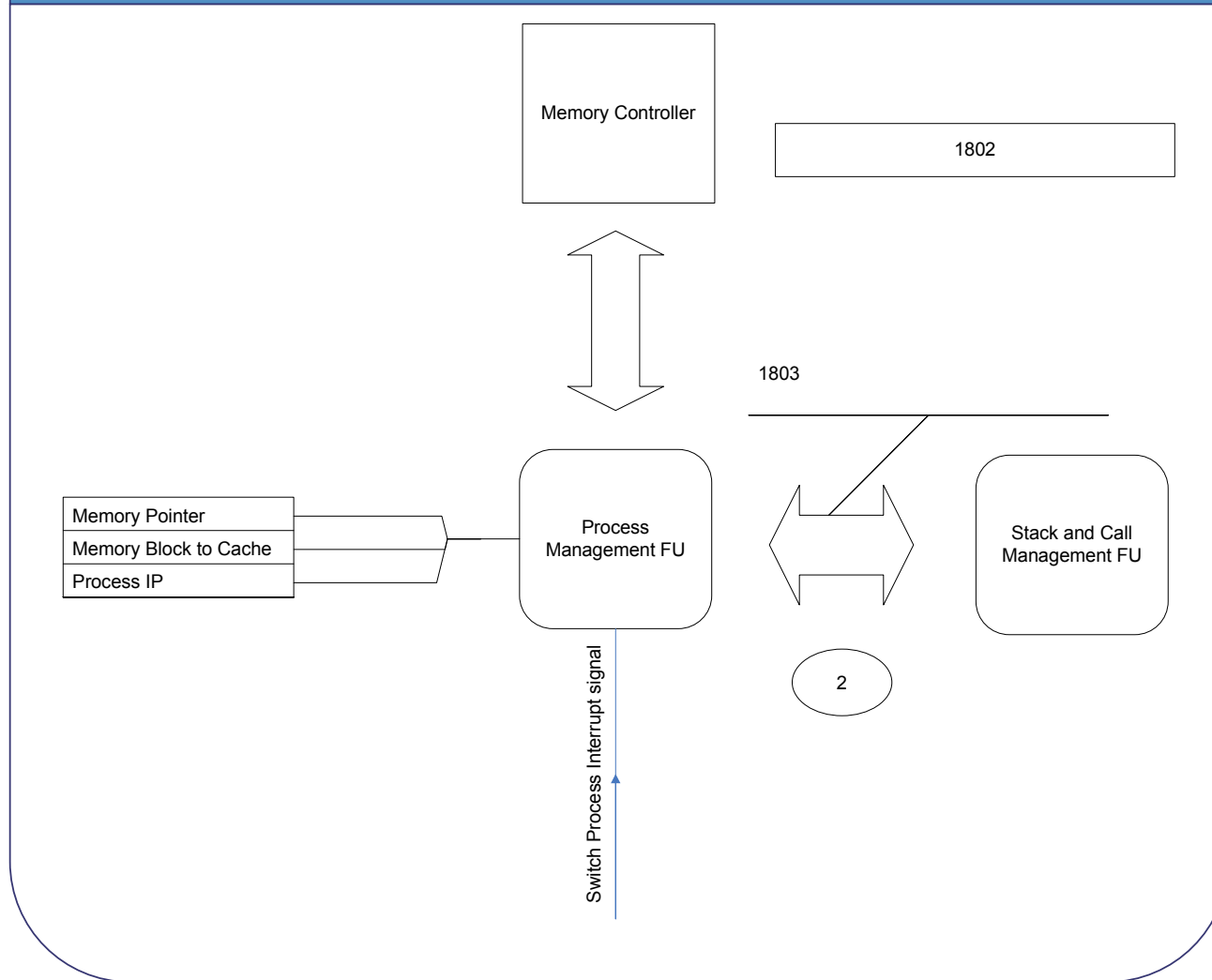


# Process Management

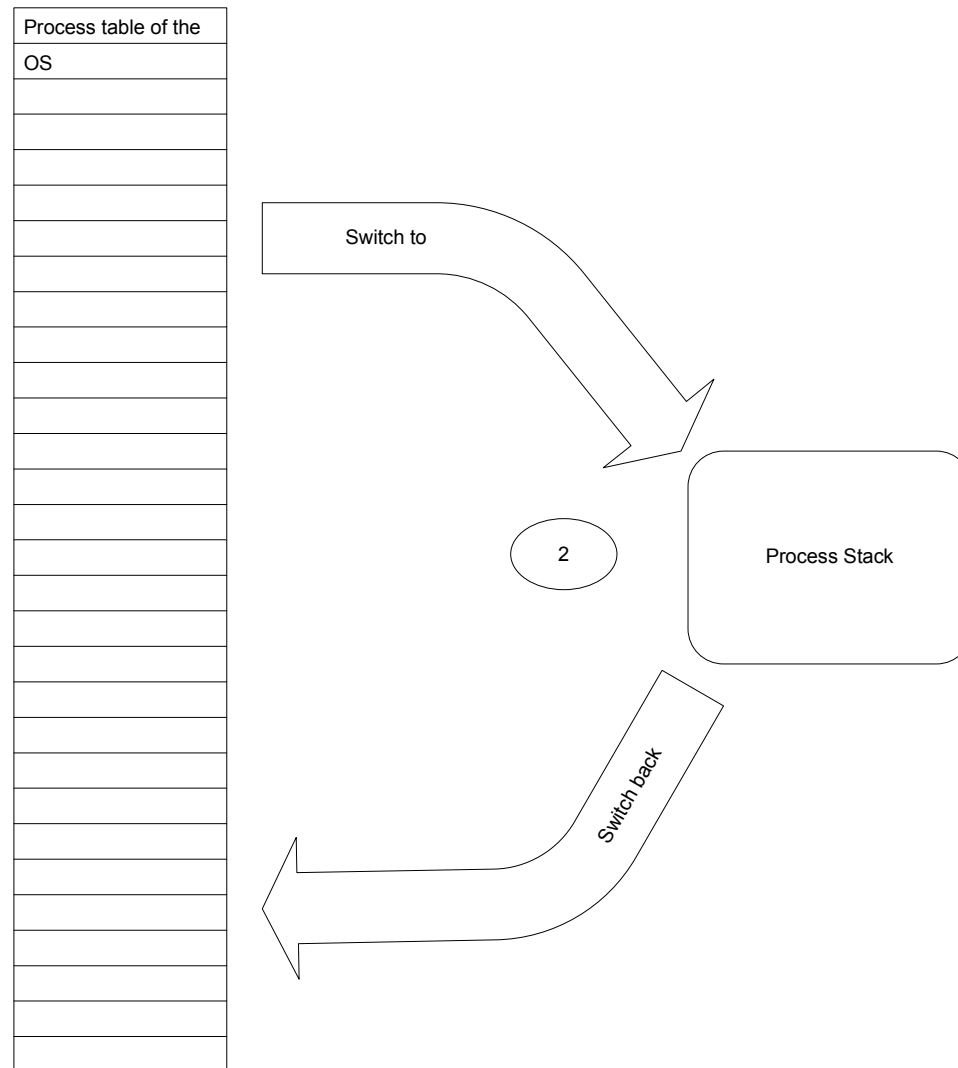
Switching Process Context and Switching

Switch to a New Process or a Previously Executed Process which has been Popped from the Stack

Switch Back to a Previously Executed Stacked Process

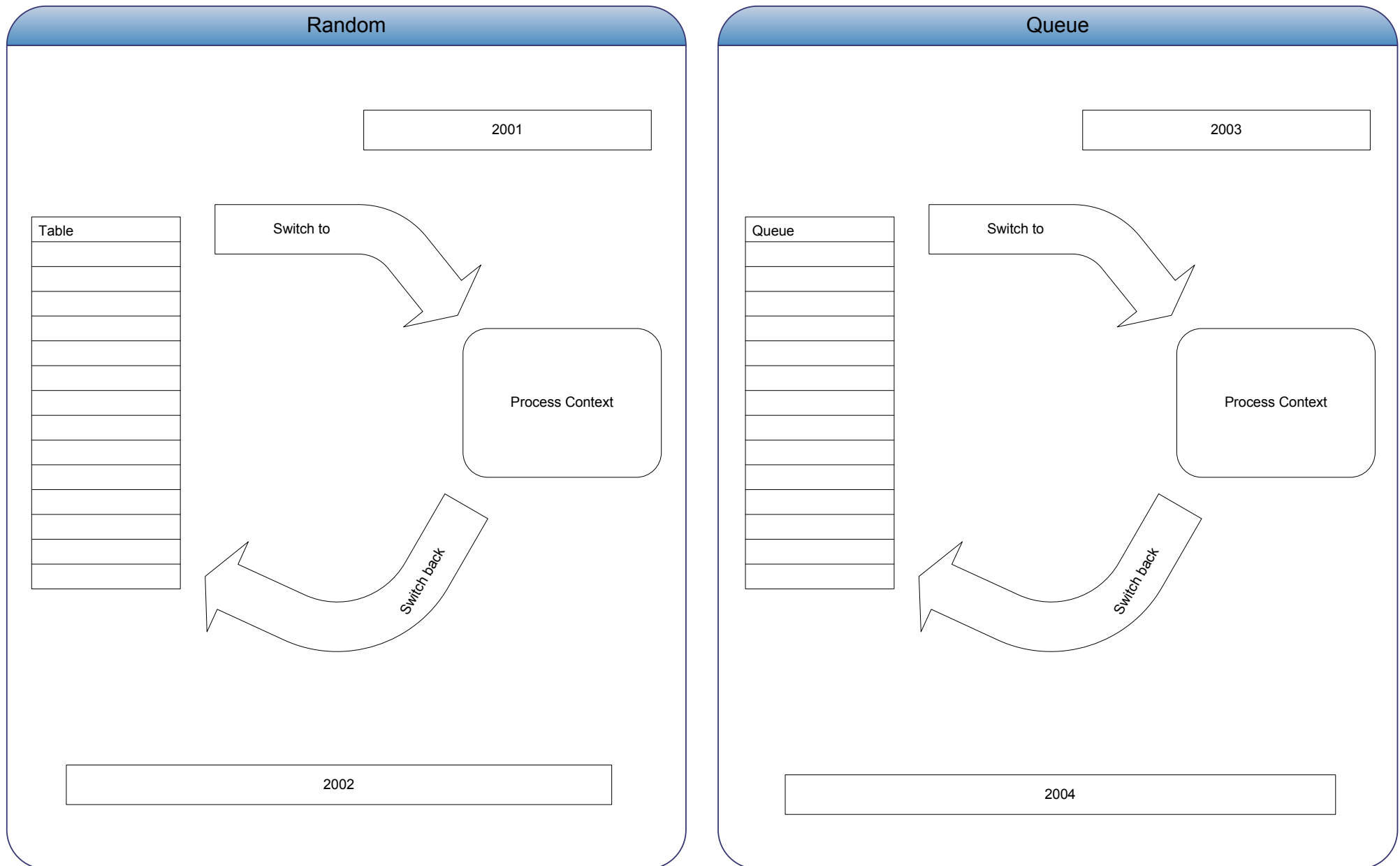


# Process Table



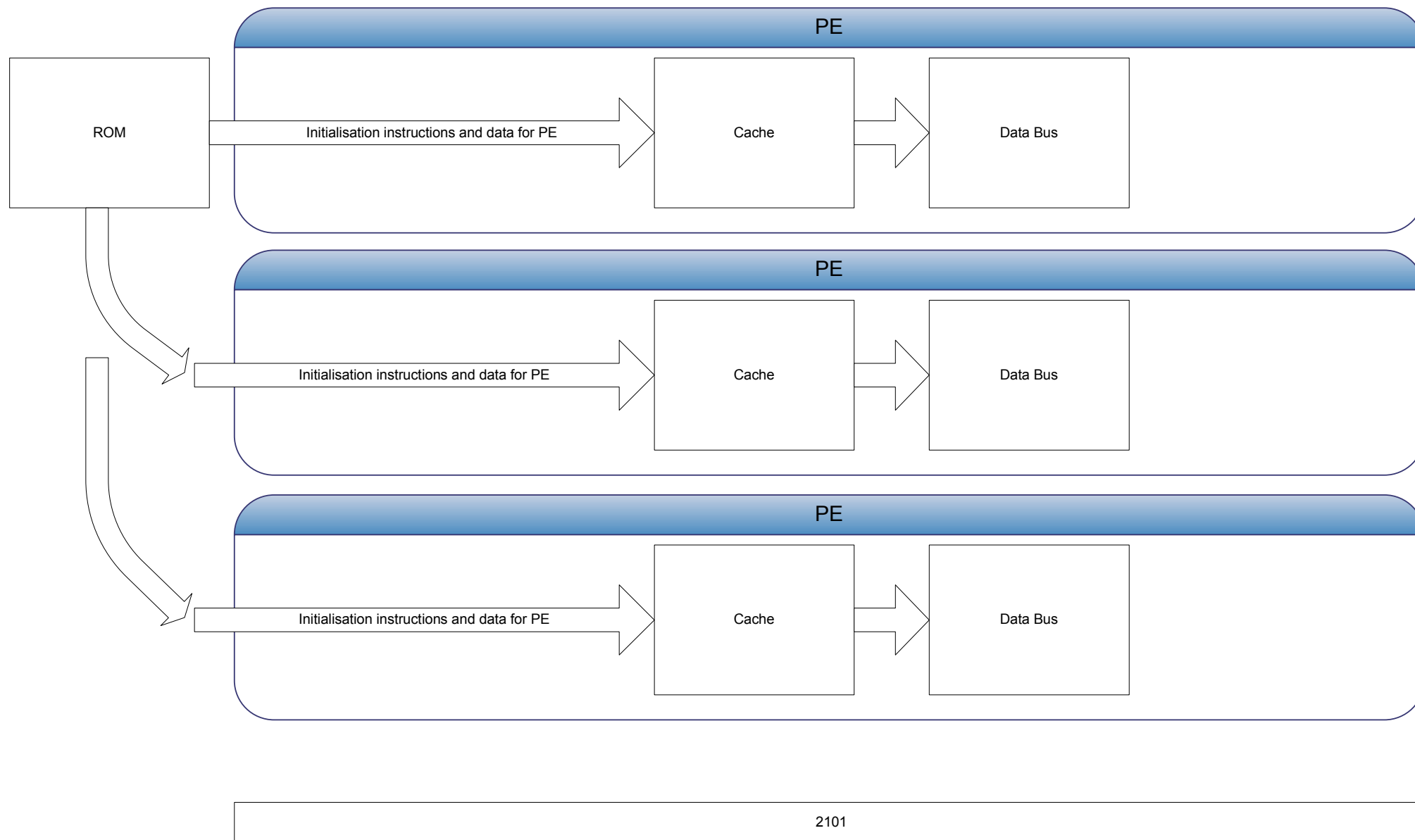
# Alternative Method For Process Management

Random and Queue Based Technique for Process Management

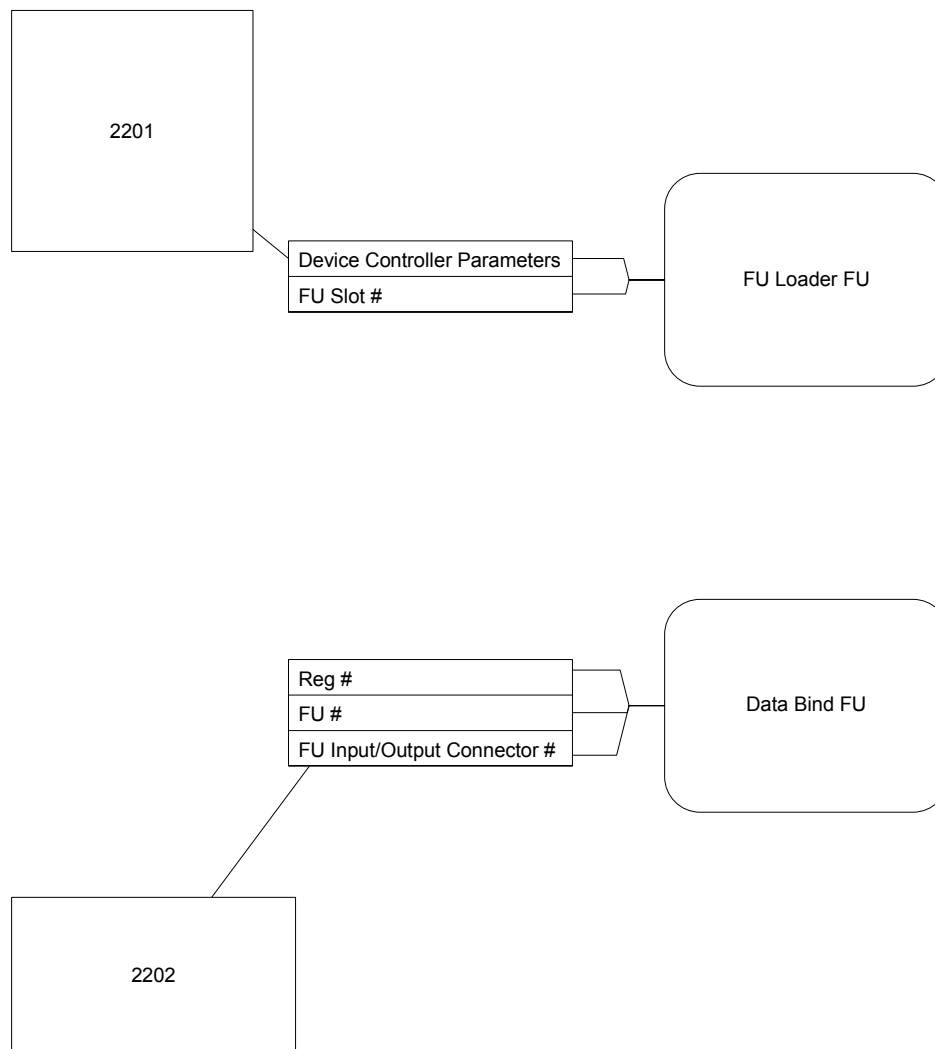


# Processor Initialization

Waking or booting up the machine



# PE Initialization



# Interrupt FU

Fig: 23 - Special FU: Interrupt FU

23/39

23 of 39 drawings

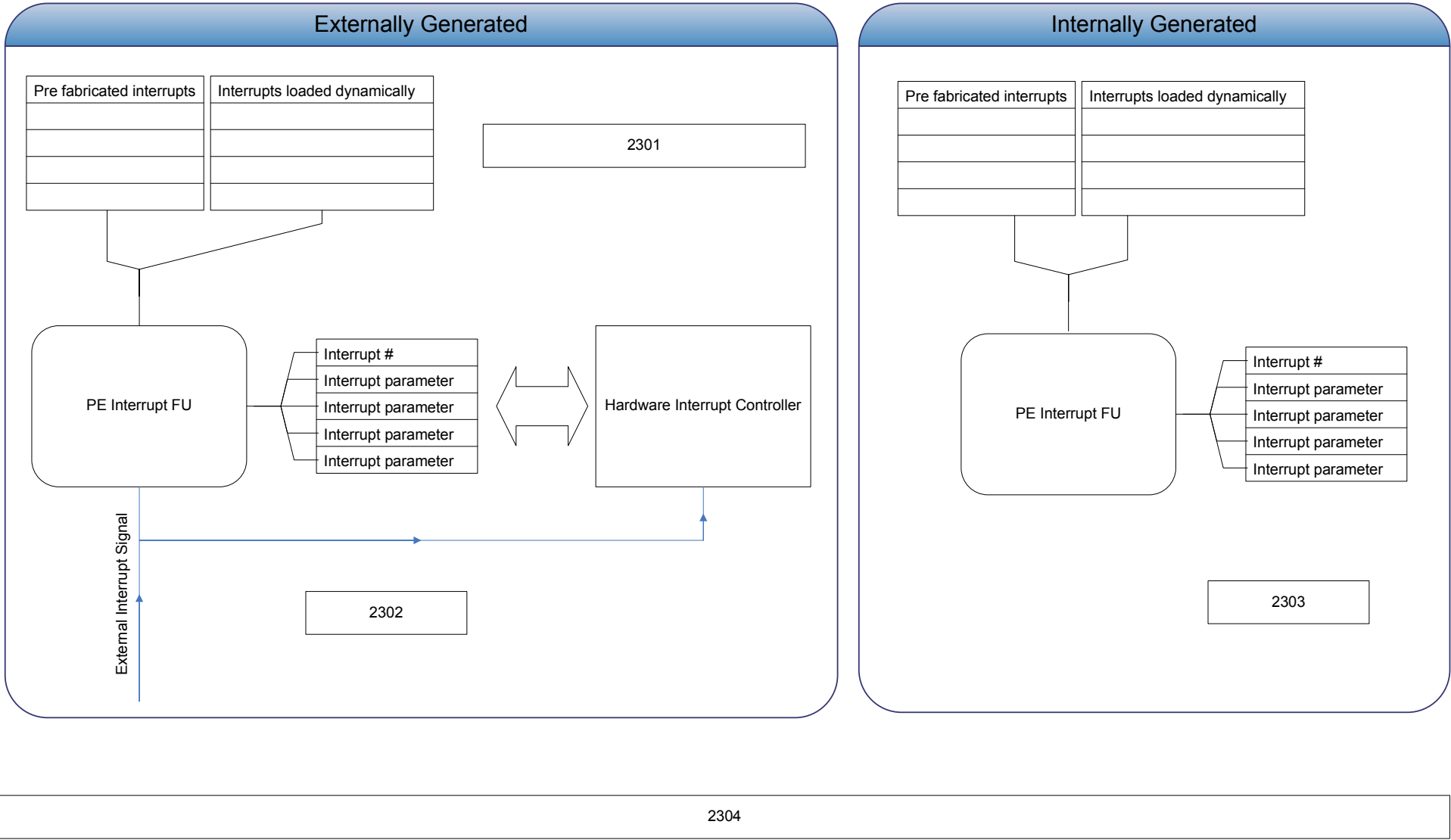


Fig: 24 - Special FU: Thread Scheduling

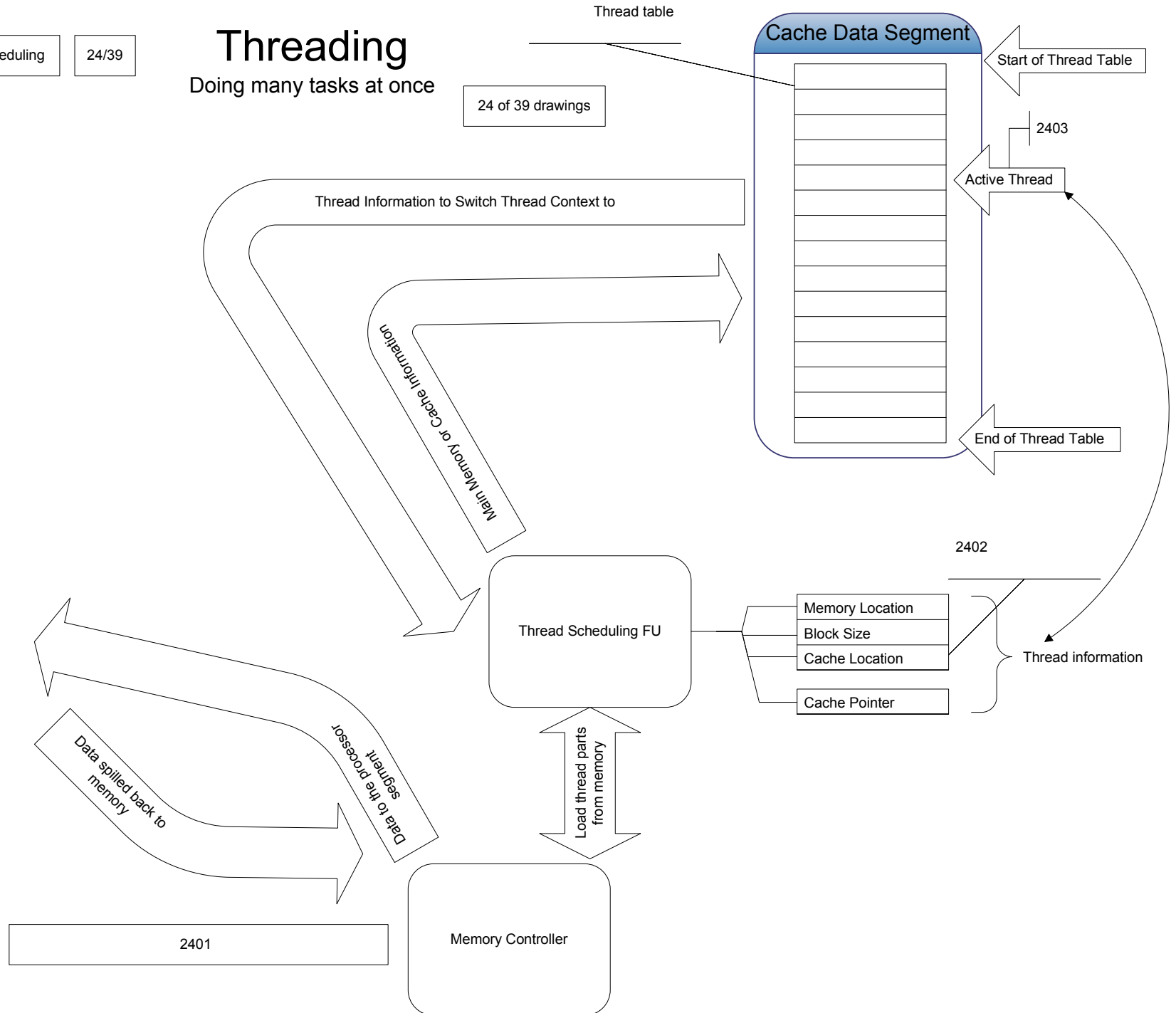
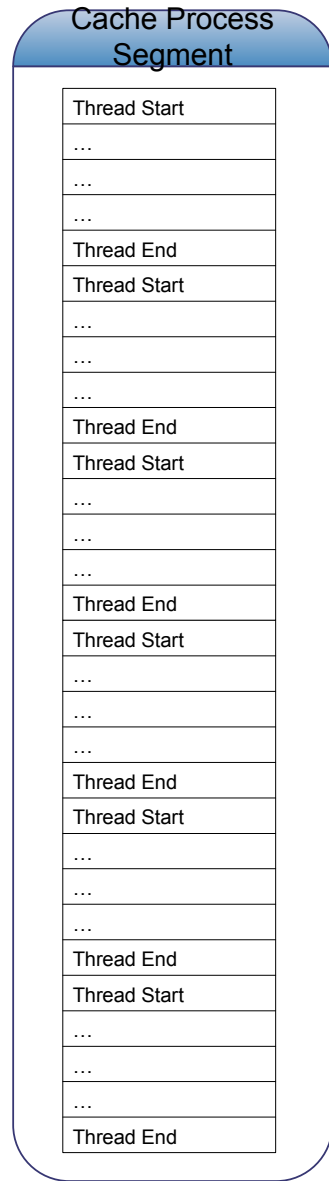
24/39

# Threading

Doing many tasks at once

24 of 39 drawings

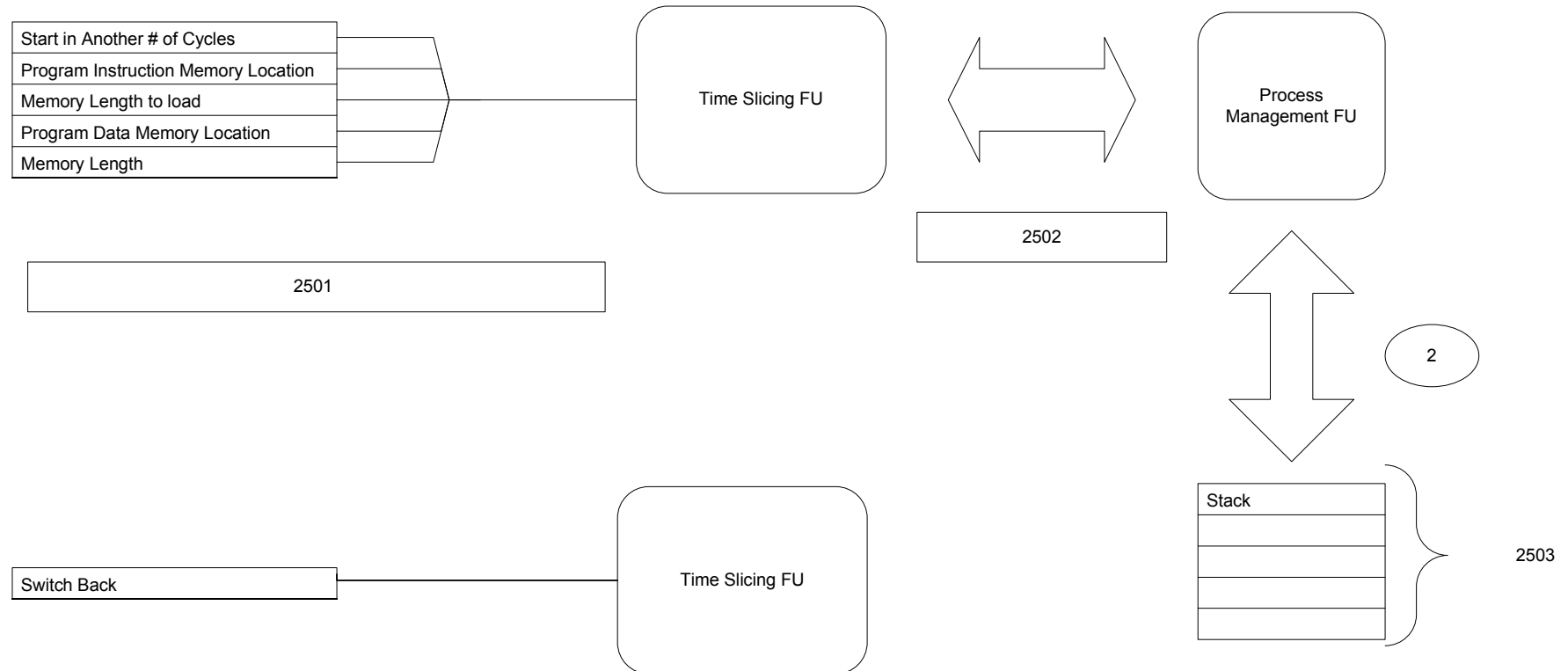
Thread table



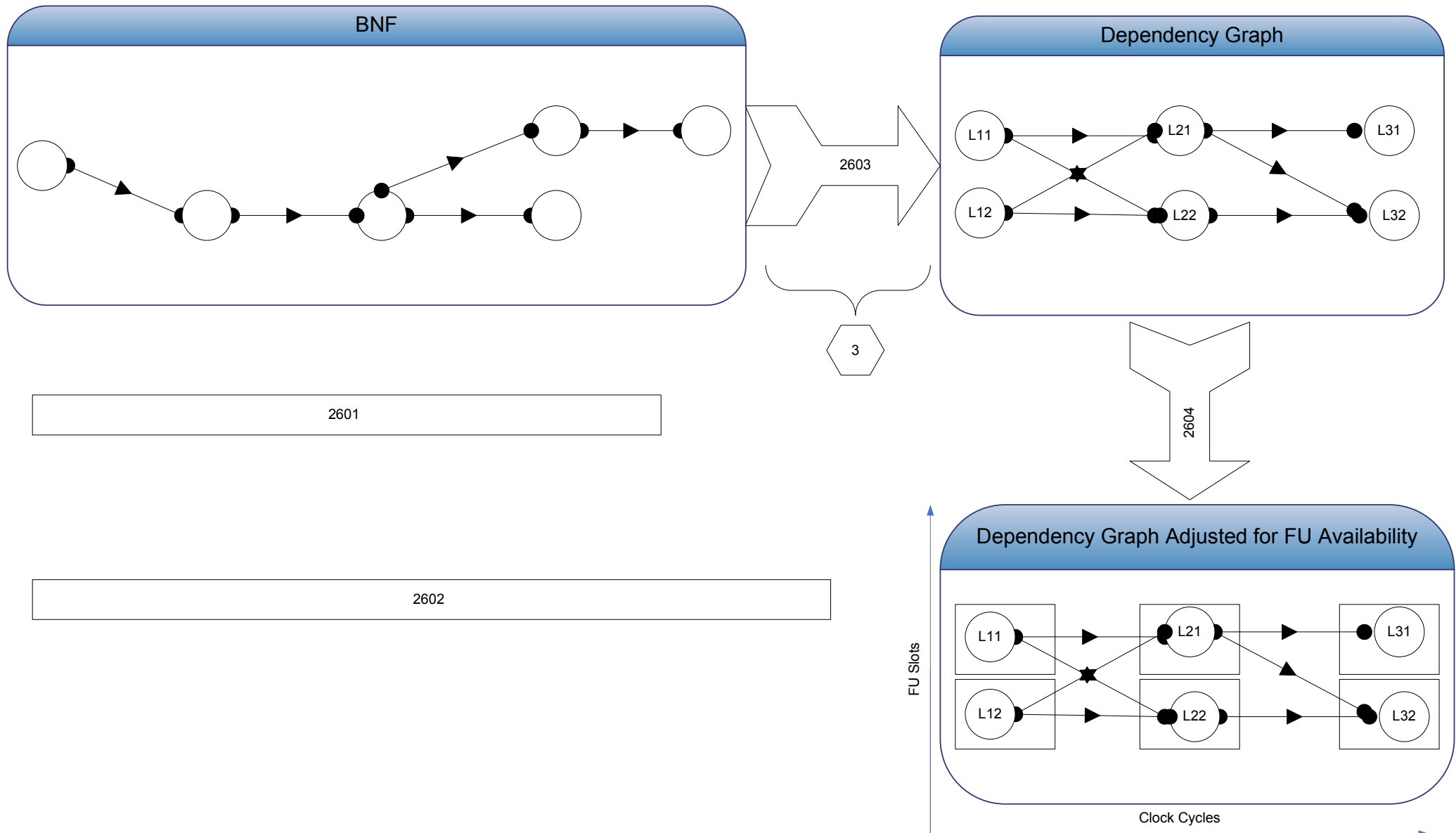


# Time Slicing FU

Running many programs at once

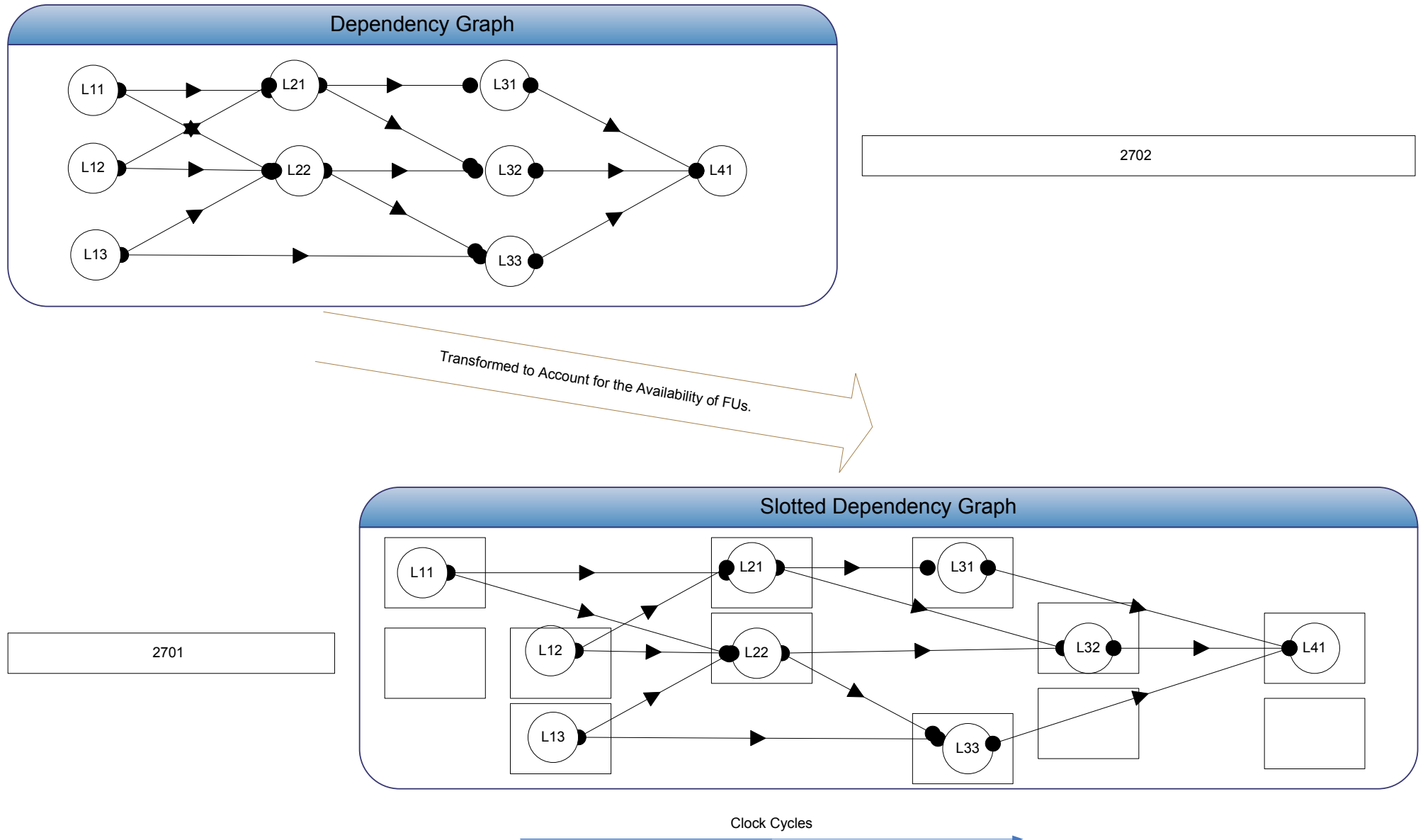


## Backus Naur Form (BNF) to Intermediate

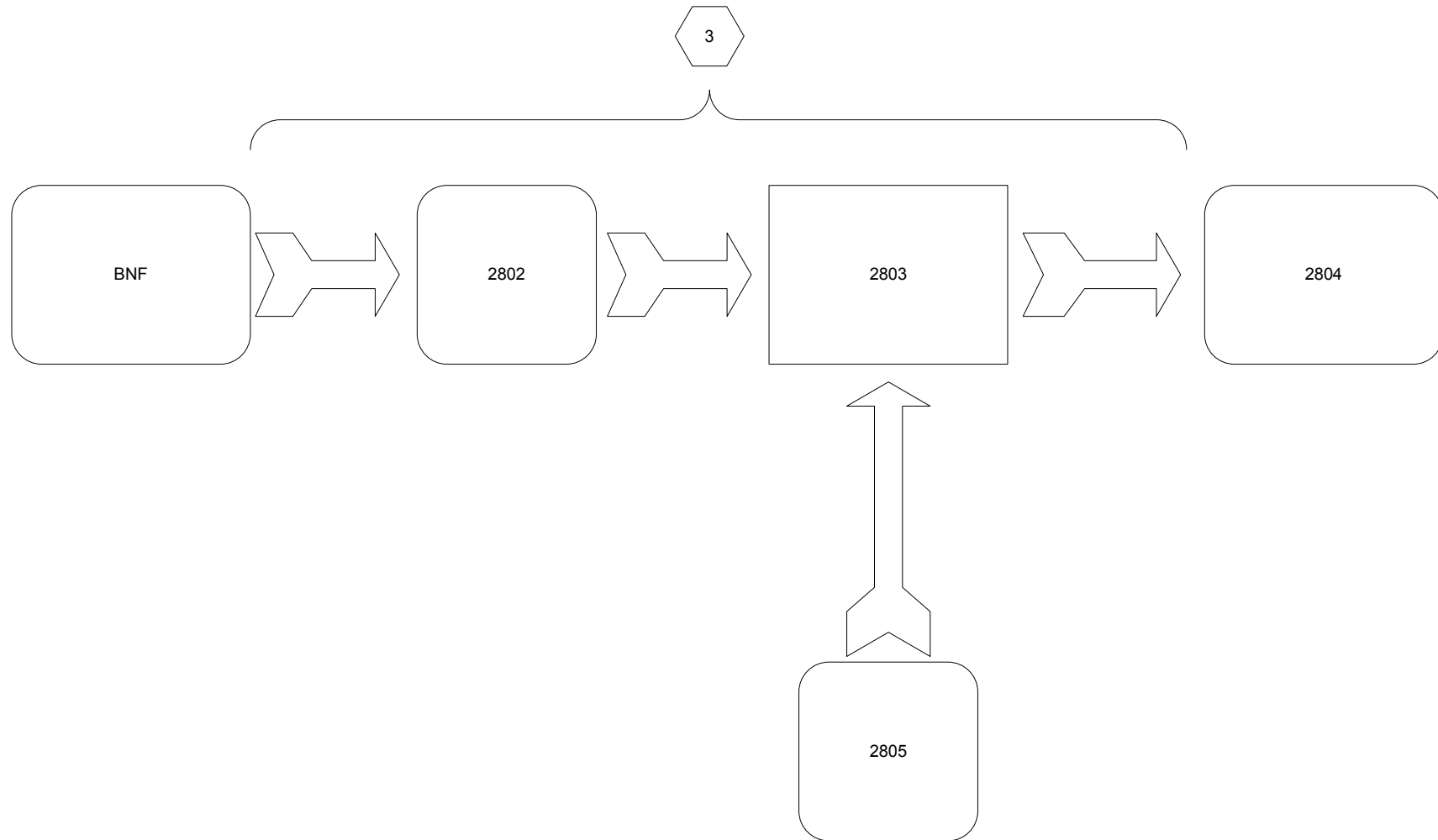


# Code Arrangement

Gaining leverage from the the parallelism of instructions

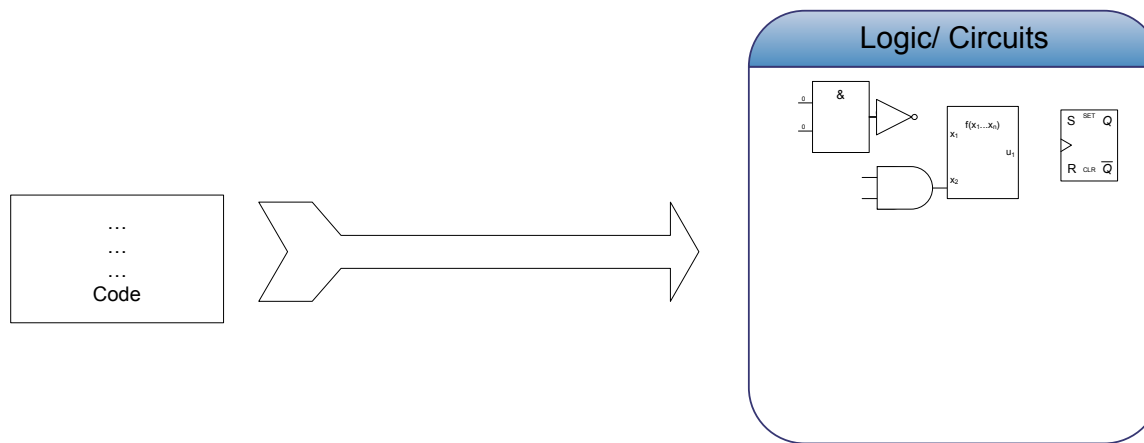


# Compiler Extensibility



# Compiling HLL to Logic

Conversion of High Level Language Code Directly into Logical Components



# Very Long Bit Registers

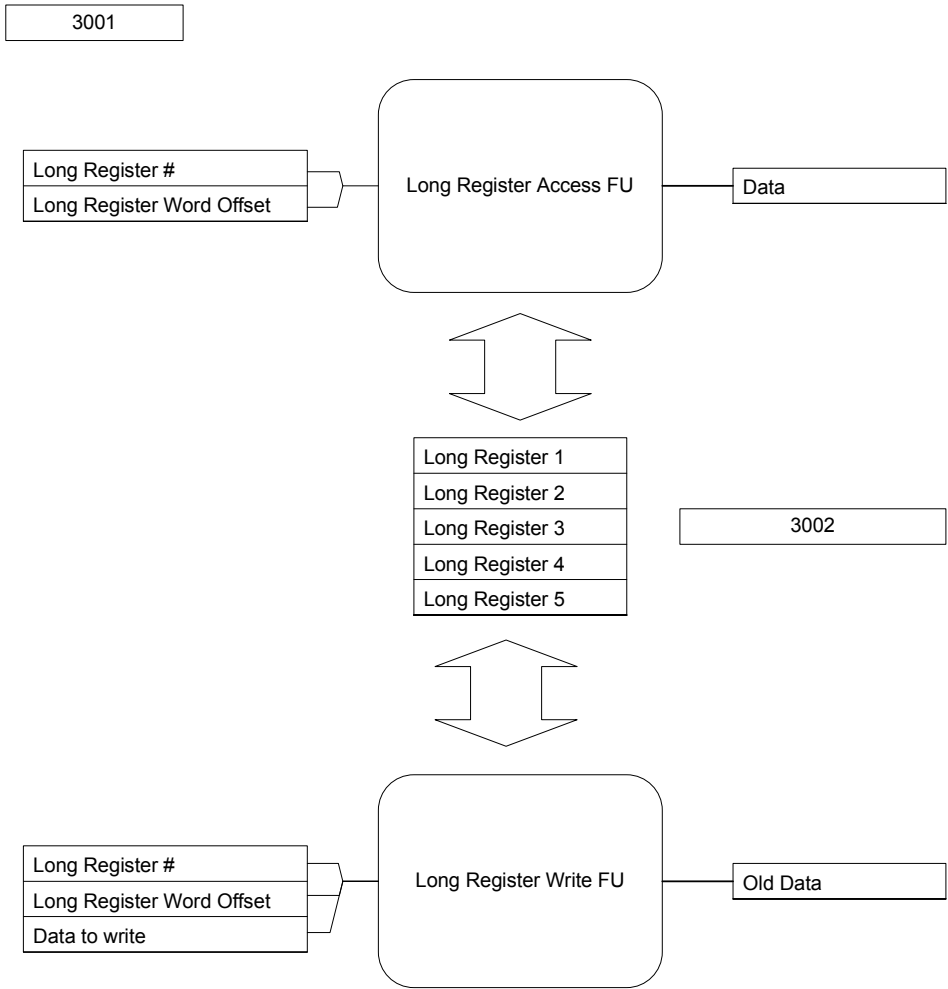
Operating on Large Chunks of Data

Fig: 30 - Array Registers

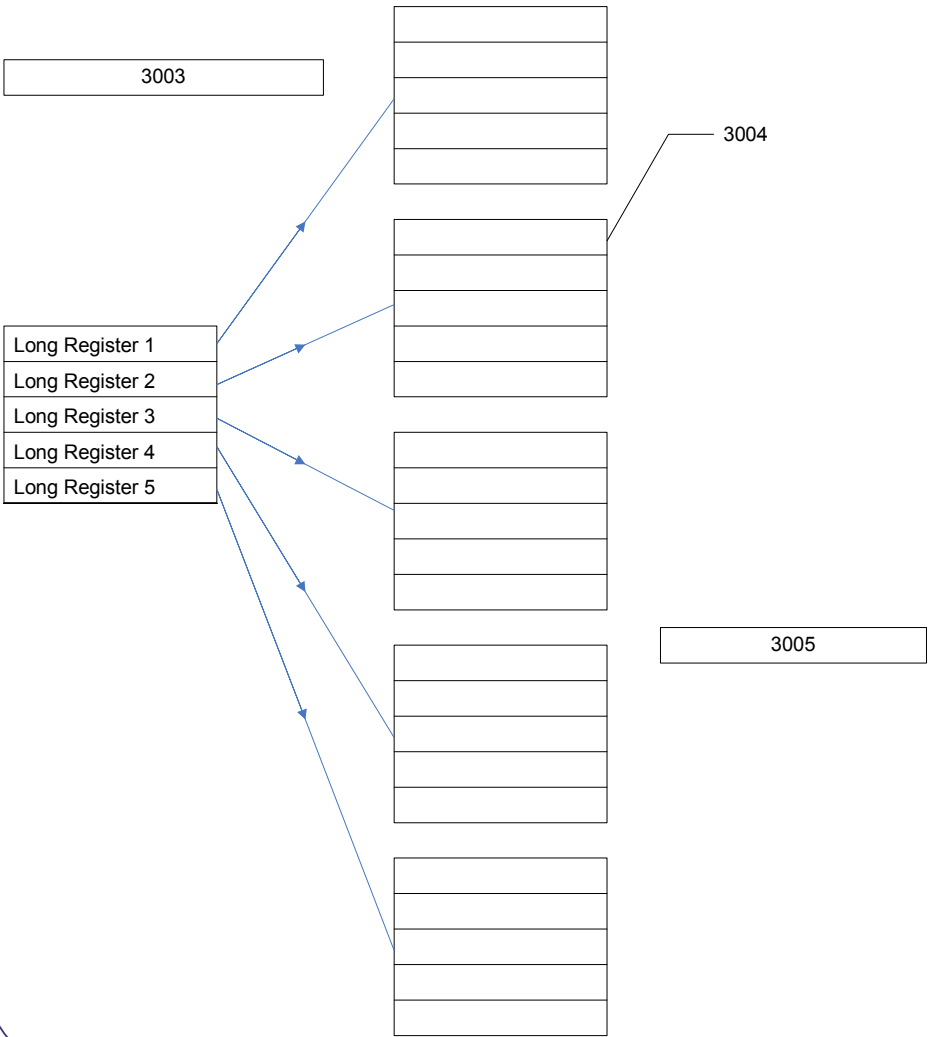
30/39

30 of 39 drawings

## Interfacing Very Long Bit Registers



## Alternate Interfacing Very Long Bit Registers



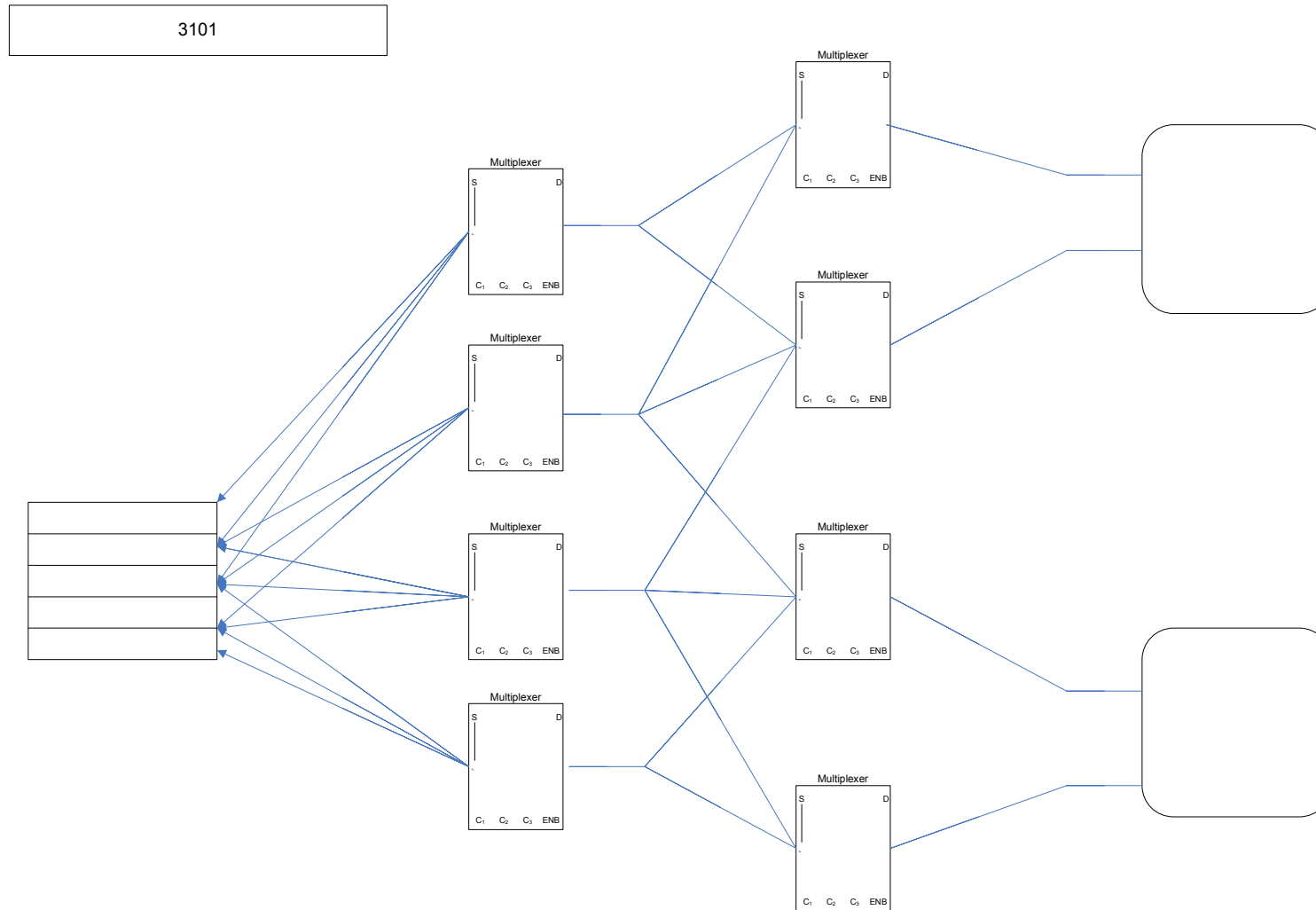
# Connecting Registers and FUs

Fig: 31 - Register FU Connectors - using mux

31/39

31 of 39 drawings

## Using Two Arrays of Multiplexers



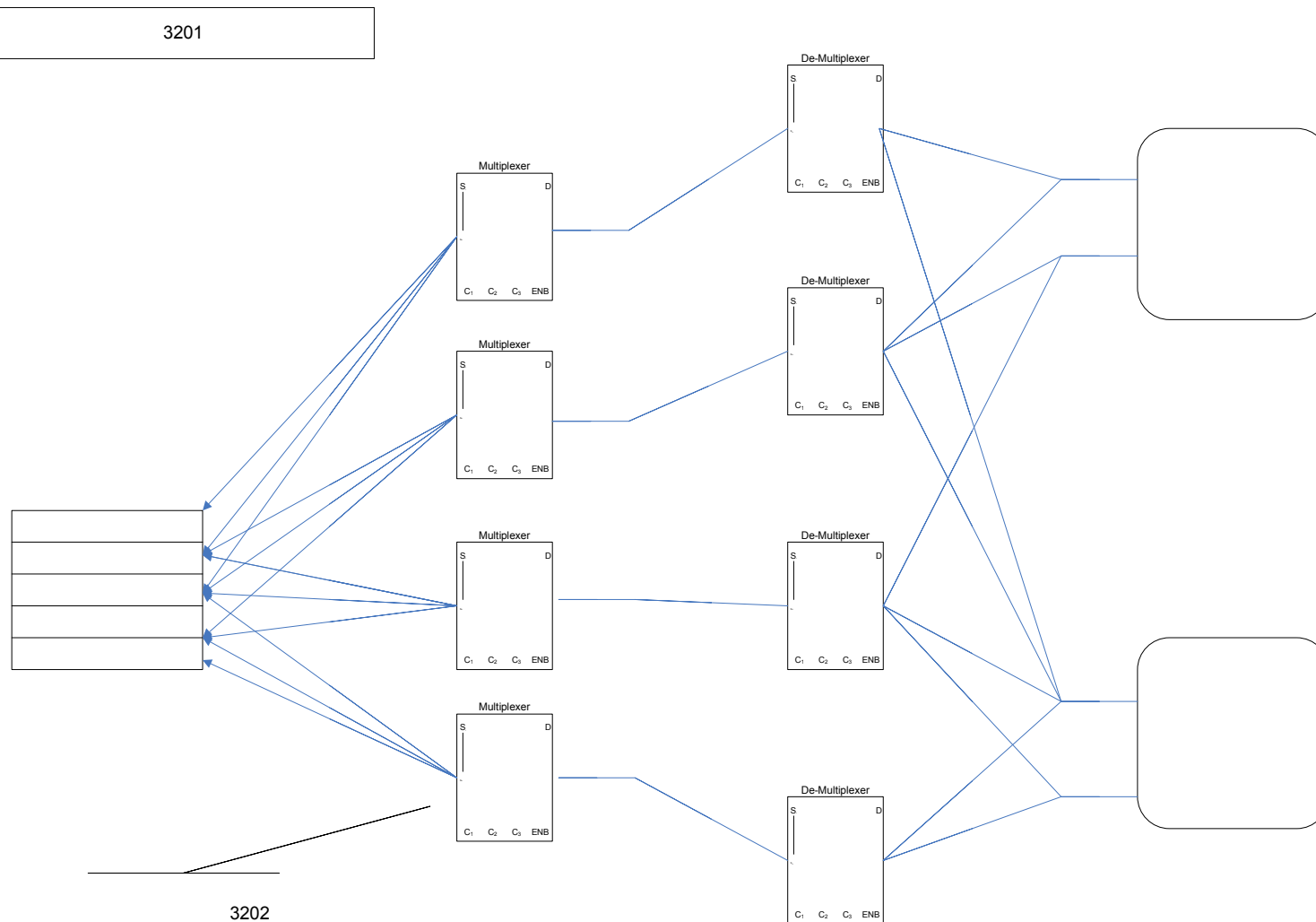
# Connecting Registers and FUs

Fig: 32 - Register FU Connections - using mux & demux

32/39

32 of 39 drawings

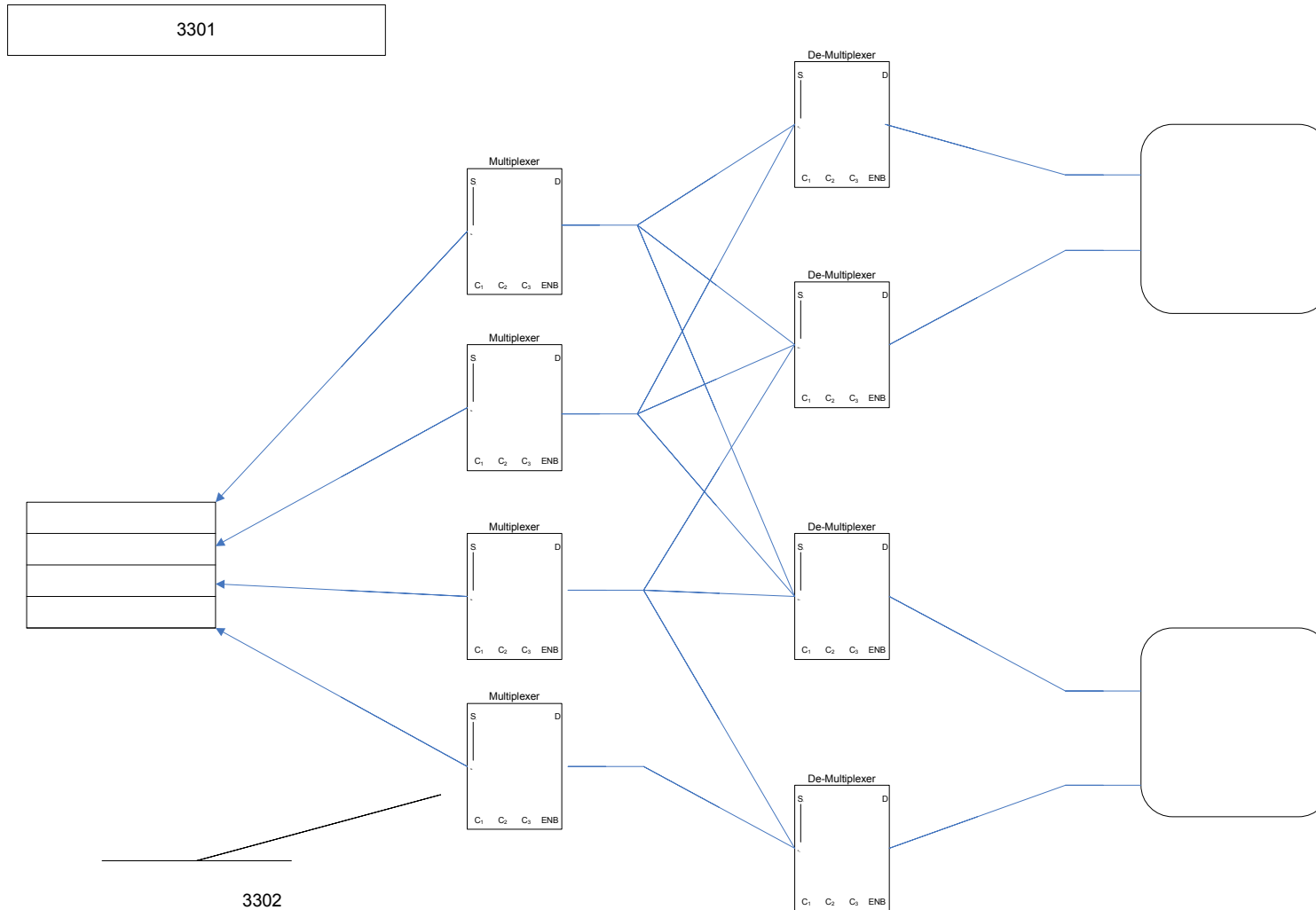
## Using Two Arrays of Multiplexes and De-Multiplexes





# Connecting Registers and FUs

## Using Two Arrays of De-Multiplexes and Multiplexes

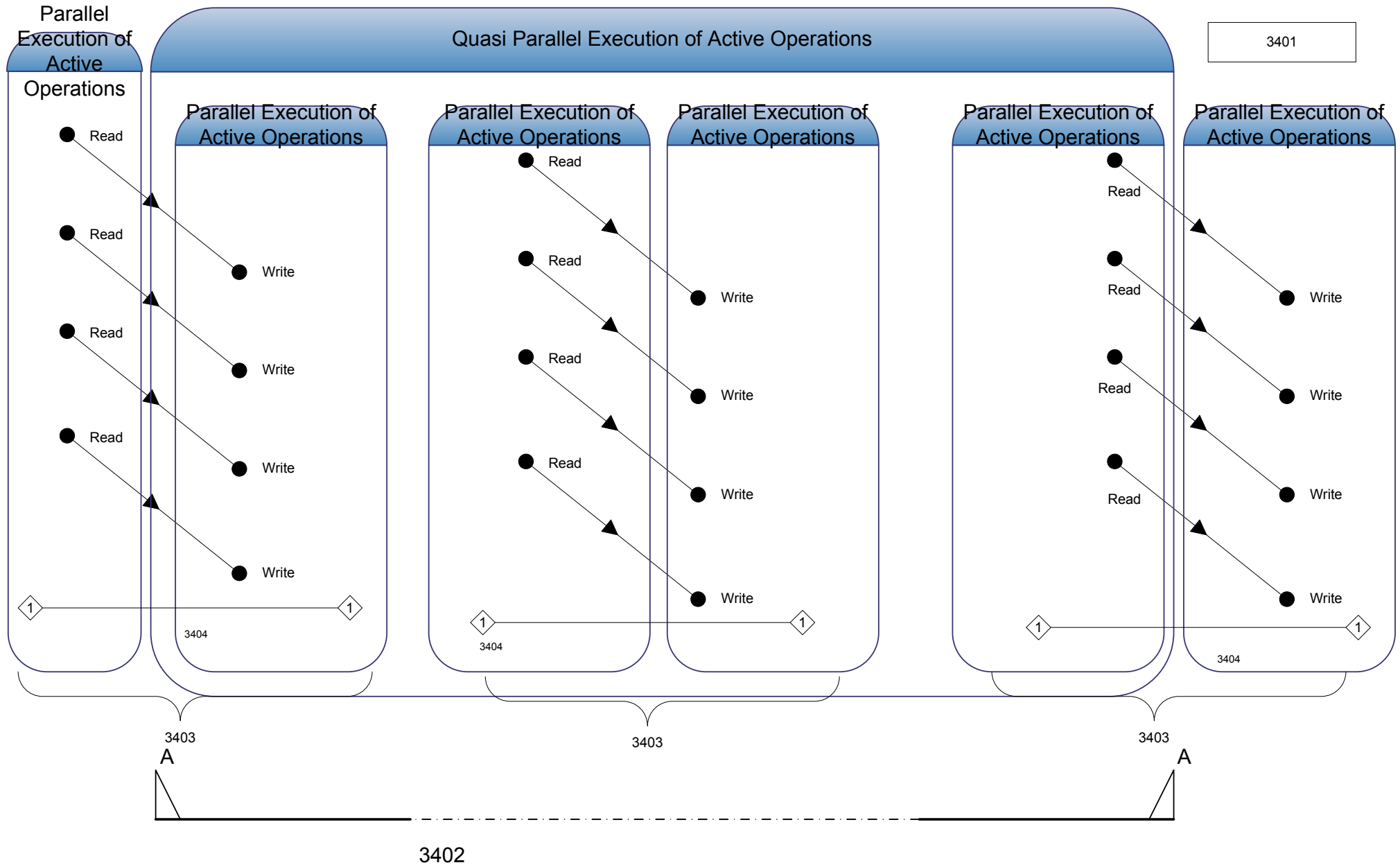


This can also be implemented using only de-multiplexes. Drawing this scheme is complex, so it is left out.

## How Active and Passive Operations Execute

34/39

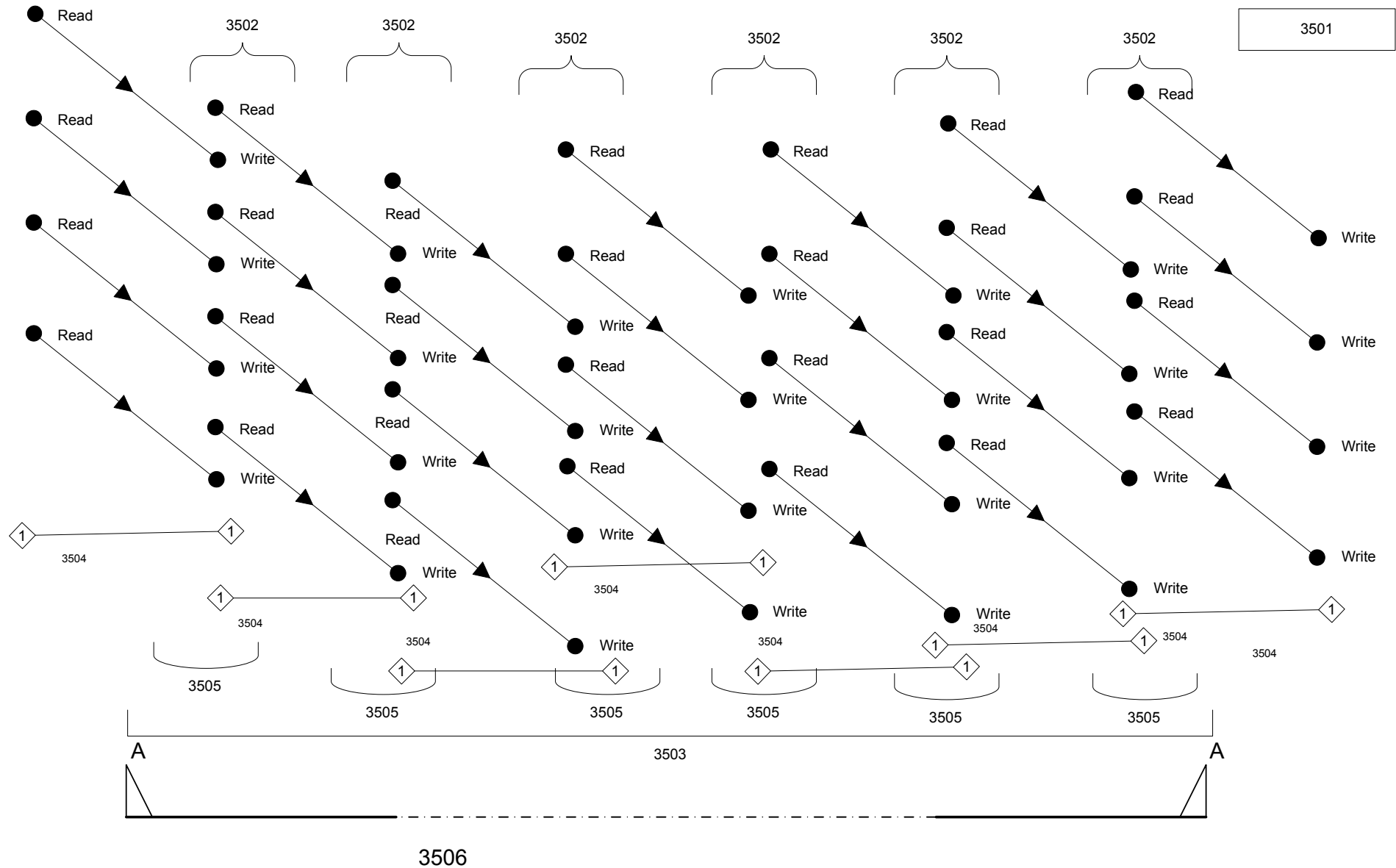
3401



## How Active and Passive Operations Execute Ideally

35/39

35 of 39 drawings



# Long Registers

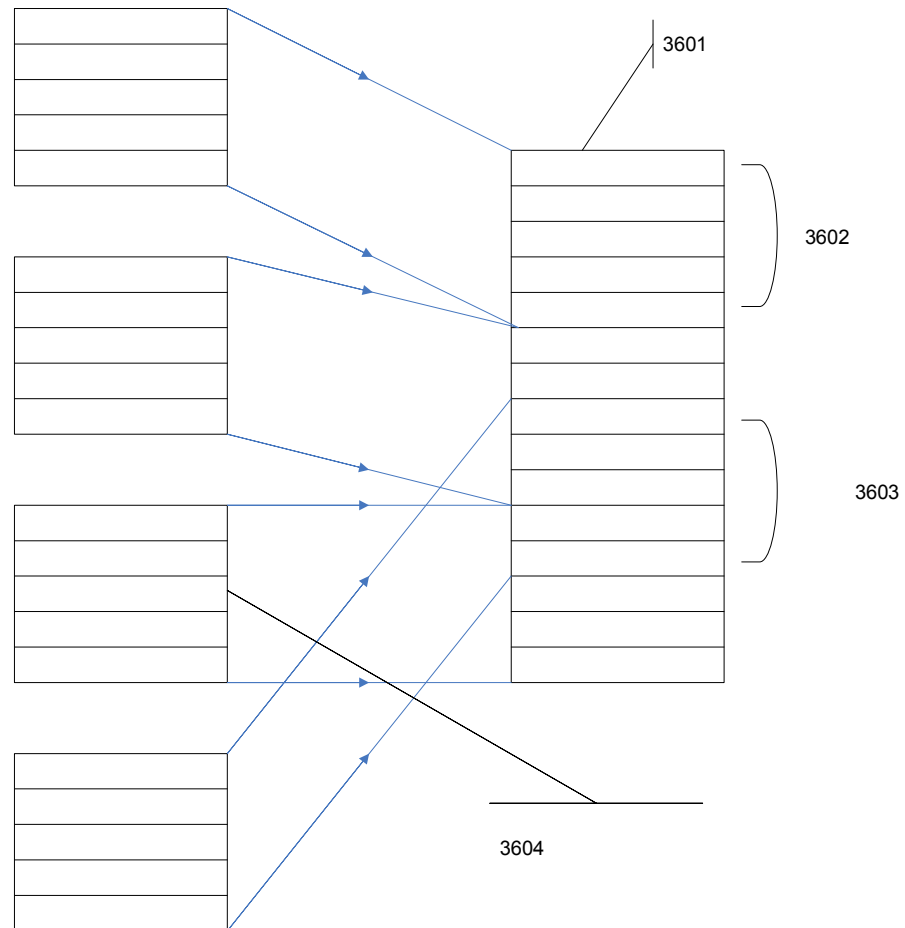
Fig: 36 - Long Registers

36/39

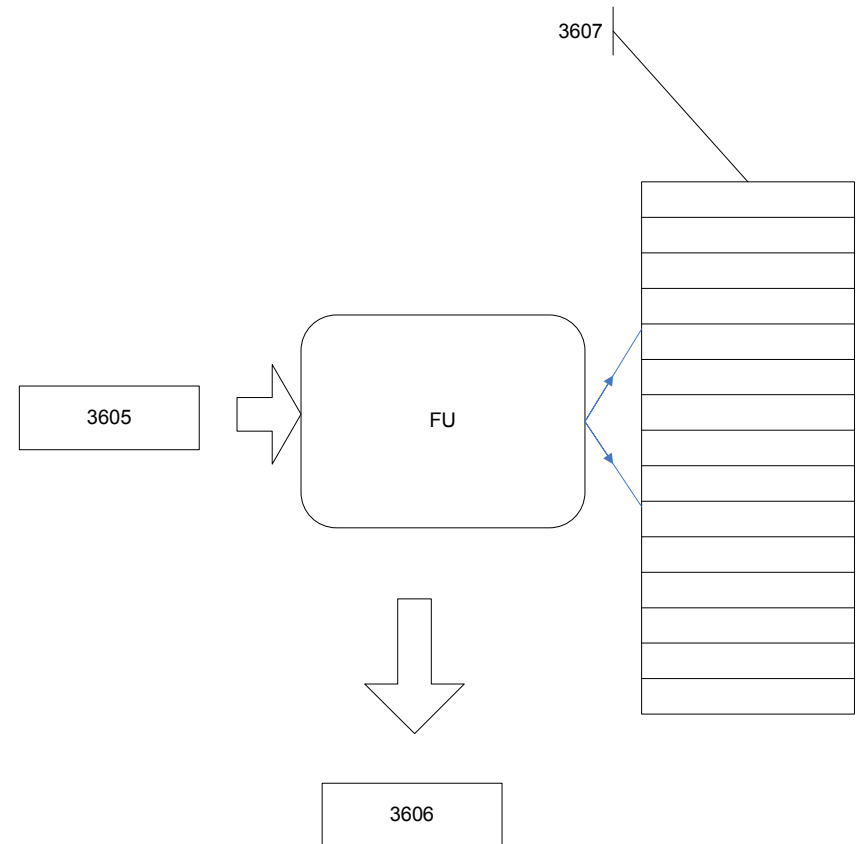
Array Registers / Bit String Registers / Byte Buffer Registers / Very Long Registers

36 of 39 drawings

Accessing Long Register through a Window



Accessing Long Register through a FU



# FU Configurations

## Configuring the FUs using Registers

Fig: 37 - FU Configuration

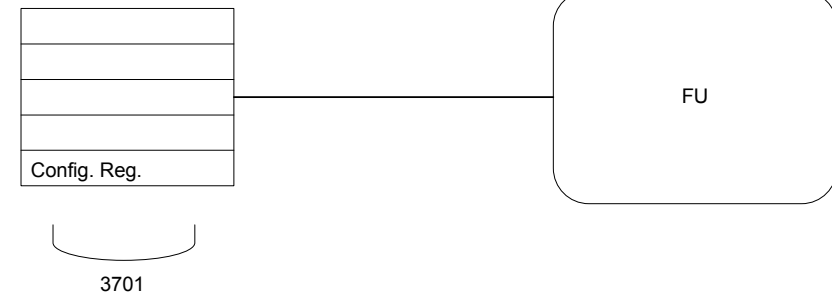
37/39

37 of 39 drawings

Configuration a FU using a complete Register



Configuring a FU using part of a Register



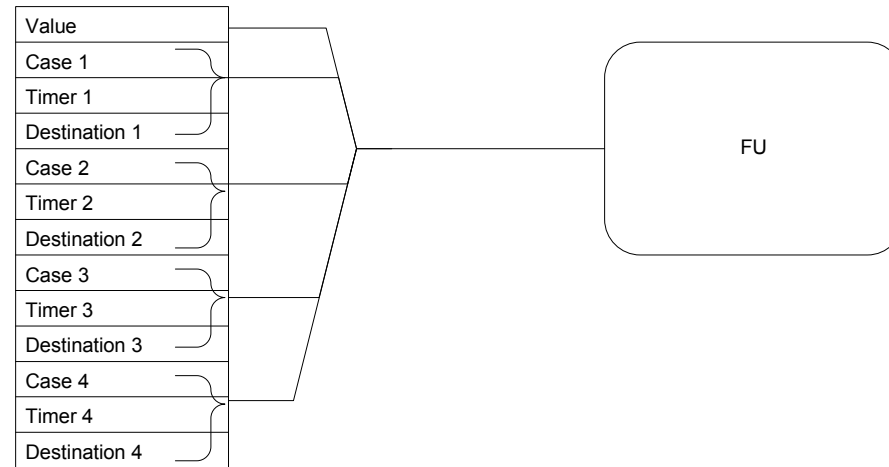
# Switch/Case FU

Selecting many Execution Path Ways

Fig: 38 - Switch

38/39

38 of 39 drawings



3801

# Symbols

## Usage and descriptions

Fig: 39 - Connection Lables

39/39

39 of 39 drawings

